Staircase Compatibility and its Applications in Scheduling and Piecewise Linearization

Andreas Bärmann[1], Thorsten Gellermann[2],
Maximilian Merkert[3] and Oskar Schneider[4]

[1]Andreas.Baermann@math.uni-erlangen.de
[2]Thorsten.Gellermann@math.uni-erlangen.de
[3]Maximilian.Merkert@math.uni-erlangen.de
[4]Oskar.Schneider@fau.de

First draft online: 6 September 2016
Revision date: 9 April 2018

Lehrstuhl für Wirtschaftsmathematik
Department Mathematik
Friedrich-Alexander-Universität Erlangen-Nürnberg
Cauerstraße 11, 91058 Erlangen, Germany

Abstract

We introduce the *Clique Problem with Multiple-Choice constraints (CPMC)* and characterize a case where it is possible to give an efficient description of the convex hull of its feasible solutions. This special case, which we name *staircase compatibility*, generalizes common properties in several applications and allows for a linear description of the integer feasible solutions to (CPMC) with a totally unimodular constraint matrix of polynomial size. We derive two such totally unimodular reformulations for the problem: one that is obtained by a strengthening of the compatibility constraints and one that is based on a representation as a dual network flow problem. Furthermore, we show a natural way to derive integral solutions from fractional solutions to the problem by determining integral extreme points generating this fractional solution. We also evaluate our reformulations from a computational point of view by applying them to two different real-world problem settings. The first one is a problem in railway timetabling, where we try to adapt a given timetable slightly such that energy costs from operating the trains are reduced. The second one is the piecewise linearization of non-linear network flow problems, illustrated at the example of gas networks. In both cases, we are able to reduce the solution times significantly by passing to the theoretically stronger formulations of the problem.

Keywords: Clique Problem with Multiple-Choice Constraints, Staircase Compatibility, Total Unimodularity, Scheduling, Piecewise Linearization

Mathematics Subject Classification: 90C27 - 90C57 - 90C35 - 90C90

1 Introduction

Compatibility structures are prevalent in many combinatorial optimization problems. In fact, they arise whenever the choice of one solution element implies the choice of other elements – in the sense of a *compatibility constraint* of the form

$$x \leq \sum_{i \in C} y_i, \tag{1}$$

with binary variables x and y_i for $C \subseteq S \coloneqq \{1, \ldots, n\}$.

In this work, we consider the combination of compatibility constraints with another frequently-occurring structure, namely so-called *multiple-choice constraints* of the form

$$\sum_{i \in S} y_i = 1. \tag{2}$$

These constraints are present whenever there is a partition of the set of eligible elements into subsets, such that it is required to choose exactly one element from each subset.

Both types of constraints together imply a clique-type problem, which can be seen by subtracting (2) from (1), resulting in a stable-set constraint on a certain graph:

$$x + \sum_{i \in S \setminus C} y_i \leq 1.$$

Generalizing this concept to an arbitrary number of compatibility and multiple-choice relations between elements leads to a very interesting problem, which we name the *Clique Problem with Multiple-Choice constraints (CPMC)*: let $G = (V, E)$ be an undirected graph and $\mathcal{V} = \{V_1, \ldots, V_m\}$ a partition of its node set V into m disjoint subsets such that each subset V_i is a stable set in G. The clique problem with multiple-choice constraints then asks to find a clique in G that contains exactly one node from each subset V_i.

While this problem is NP-hard in general (see [BGM18] or [Mer17]), we want to investigate here a special case where it is solvable in polynomial time. This is possible for a restriction of the compatibility graphs to graphs with a certain compatibility structure which we name *staircase compatibility*. For clique problems exhibiting this special structure, we will be able to state efficient linear programming (LP) formulations of which we can show that the corresponding constraint matrix is totally unimodular.

In order to demonstrate that there is great benefit from studying this structure, we present two very distinct real-world applications which can be modelled as (CPMC) under staircase compatibility. The first one is a problem in railway timetabling which falls into the class of project scheduling problems. We will see that the precedence constraints of the project scheduling problem have a very natural correspondence to the compatibility constraints in (CPMC). The second application arises in the context of piecewise-linear approximation of non-linear functions in optimization problems on transport networks. In both cases, the resulting model reformulations are already known (see [MSSU01] and [LM16] respectively). However, our notion of staircase compatibility provides a common, more general framework to study the underlying clique problem with multiple-choice constraints. In particular, we are able to show that the derived integrality results hold for a wider class of compatibility graphs.

We begin with the definition of staircase compatibility in Section 2, which is accompanied by a first discussion of its presence in project scheduling problems and flow

problems with a piecewise-linear objective function as well as similar structures in the literature. Following that, we derive two totally unimodular LP formulations for the clique problem with multiple-choice constraints in the case of staircase compatibility in Section 3. The second of these two reformulations takes the form of a dual network flow problem. It will give rise to a very natural way of generating heuristic solutions from a fractional solution to the problem by determining integral extreme points which generate this fractional solution. In Section 4, we present our computational results for the two practical applications mentioned above, showing that the better understanding of their staircase structure directly translates into vastly shorter solution times. Finally, in Section 5, we summarize the findings of this paper and give possible directions for an extension of our results.

2 Staircase Compatibility

In the following, we define the clique problem with multiple-choice constraints as it is considered here as well as the notion of staircase compatibility. We will see later that the presence of such staircase structures in the problem allows for an efficient description of the convex hull of its feasible solutions.

Definition 2.1 (The Clique Problem with Multiple-Choice Constraints)**.** *Let $G = (V, E)$ be an undirected graph and $\mathcal{V} = \{V_1, \dots, V_m\}$ a partition of V such that each $V_i \in \mathcal{V}$ is a stable set in G. The* clique problem with multiple-choice constraints *(CPMC) is then given by the task to*

$$\text{choose exactly one node from each subset } V_i \tag{Rule 1}$$

such that the selected nodes form a clique in G. An instance of (CPMC) is consequently given by the pair $(G, \mathcal{V})$, where we refer to G as the compatibility graph *of the (CPMC)-instance.*

In this work, we focus on a special case of (CPMC) with a certain structure in the edge set of the underlying graph:

Definition 2.2 (Staircase Compatibility)**.** *Let $(G, \mathcal{V})$ be an instance of (CPMC) and let each subset V_i in $\mathcal{V}$ be ordered according to a total order $<_i$. The latter allows us to denote the elements of V_i by $v_{i,1}, \dots, v_{i,n_i}$ with $n_i = |V_i|$ when we directly want to refer to this order. In the following, we omit the index i and simply write $<$ whenever no confusion is possible. We then call the instance $(G, \mathcal{V})$* staircase *if the following two conditions hold. The first condition implies the connectedness of the compatible choices for a given element:*

$$\{v, w_1\} \in E \wedge \{v, w_3\} \in E \Rightarrow \{v, w_2\} \in E, \tag{Rule 2a}$$

whenever $v \in V_i$, $w_1, w_2, w_3 \in V_j$ with $w_1 < w_2 < w_3$. The second condition enforces a certain kind of monotonic behaviour of E:

$$\{v_1, w_2\} \in E \wedge \{v_2, w_1\} \in E \Rightarrow \{v_1, w_1\} \in E \wedge \{v_2, w_2\} \in E \tag{Rule 2b}$$

for $v_1, v_2 \in V_i$, $w_1, w_2 \in V_j$ with $v_1 < v_2$ and $w_1 < w_2$.

Our choice of the term "staircase compatibility" becomes clear when considering the adjacency matrix corresponding to the compatibility graph: each submatrix that describes the compatibility between the nodes of two subsets of the partition is a staircase matrix if its rows and columns are ordered according to the $<_i$ (see [Fou84] for an

extensive compilation of the properties of staircase matrices). In short, such a submatrix will have a consecutive block of 1's in each row, where the first 1 in each row has a column index at least as high as that of the first 1 in the preceding row, and the same for the respective last 1's in two successive rows.

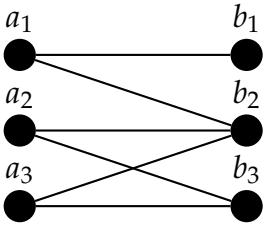
Note that we assume in this article that each node in a subset $V_i \in \mathcal{V}$ has at least one node in each of the remaining subsets with which it is compatible. Otherwise, this node may be eliminated as it cannot belong to a feasible selection. Note also that in this case (Rule 2b) implies (Rule 2a).

The above definitions prepare us to consider the problem of interest in this paper:

Definition 2.3 ((CPMC) under Staircase Compatibility)**.** *If the instance $(G, \mathcal{V})$ of (CPMC) is staircase, then we name the arising special case of (CPMC) the* clique problem with multiple-choice constraints under staircase compatibility (CPMCS).

Before we present interesting applications and a further discussion of this problem, we give an example which is taken up again in Section 3.

Example 2.4. *Consider the following example:*



$$\begin{array}{c} \\ a_1 \\ a_2 \\ a_3 \end{array} \begin{array}{c} \begin{array}{ccc} b_1 & b_2 & b_3 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix} \end{array}$$

It shows the compatibility graph for (CPMCS) as well as the corresponding adjacency matrix for a certain instance $(G, \mathcal{V})$ where $V = V_1 \cup V_2$ with $V_1 = \{a_1, a_2, a_3\}$ and $V_2 = \{b_1, b_2, b_3\}$. We see that removing edge $\{a_2, b_2\}$ would invalidate (Rule 2a) *(and also* (Rule 2b)*, see above), while removing $\{a_3, b_3\}$ would invalidate* (Rule 2b)*. Any selection $\{a, b\}$ with $\{a, b\} \in E$ would be feasible for (CPMCS).*

2.1 Two Indicative Examples of (CPMCS)

In the following, we give two example applications where (CPMCS) is present as a substructure. In both examples, the project scheduling problem and interval compatibilities in path flows, it is a possible way to model the set of feasible solutions.

Project Scheduling Consider m tasks $i = 1, \ldots, m$. Each task i has to be assigned exactly one starting time, where we assume that there is a finite discrete set $T_i = \{t_{i,1}, \ldots, t_{i,n_i}\} \subset \mathbb{R}$ of possible starting times that may differ for different tasks. Additionally, pairs of tasks may have precedence relations, requiring one of the two to start in a predefined time window relative to the other one (if no precedence relations are given, they may be done in any order, or possibly in parallel). This problem is called the *project scheduling problem with precedence constraints*. For further information and examples see [SZ15] and the references therein.

The following is a possible formulation for the problem of finding a feasible sched-

ule in the above setting:

$$\begin{aligned} \text{find} \quad & x \\ \text{s.t.} \quad & x_k - x_l \geq \underline{d}_{kl} \quad 1 \leq k < l \leq m && \text{(3a)} \\ & x_k - x_l \leq \bar{d}_{kl} \quad 1 \leq k < l \leq m && \text{(3b)} \\ & x_i \in T_i \quad i = 1, \ldots, m && \text{(3c)} \end{aligned}$$

for some $\underline{d}_{kl}, \bar{d}_{kl}$ with $k = 1, \ldots, m$ and $l = k+1, \ldots, m$.

We can model this problem as (CPMCS) as follows: each subset V_i represents a task i, where the elements in each subset are identified with the possible starting times $\{t_{i,1}, \ldots, t_{i,n_i}\}$. Consequently, the subsets come with a natural chronological ordering. Furthermore, for different tasks k, l with $k \neq l$, we have

$$\{t_{k,j_k}, t_{l,j_l}\} \in E \Leftrightarrow \underline{d}_{kl} \leq t_{k,j_k} - t_{l,j_l} \leq \bar{d}_{kl}$$

for all $1 \leq j_k \leq n_k$ and $1 \leq j_l \leq n_l$. It can easily be seen that (Rule 2a) is satisfied due to the convexity of the relative time window defined by $\underline{d}_{kl}$ and $\bar{d}_{kl}$. Furthermore, violating (Rule 2b) would contradict the temporal ordering. Therefore, $G = (V, E)$ and $\mathcal{V}$ as defined here define a staircase instance.

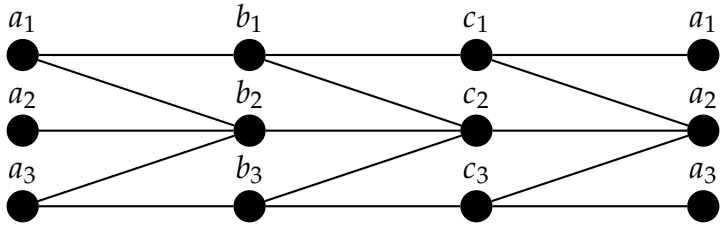
Interval Compatibilities in Path Flows Consider a path-shaped network consisting of m edges $e_1, \ldots, e_m$. Each edge has an interval for the feasible flow on that edge which furthermore is subdivided into n_i subintervals, $i = 1, \ldots, m$. This scenario appears as a substructure in network flow problems where the flow cost has been piecewise linearized (see Section 4.2 for more details; cf. also [LM16]). The task is to describe the set of feasible combinations of flow intervals.

It represents a special case of (CPMCS) as can be seen as follows: define V as the set of all intervals, where subset V_i includes all intervals belonging to edge e_i on the path. As these intervals are obtained from subdividing a larger interval, a canonical ordering is available. Intervals belonging to different (not necessarily adjacent) edges are *compatible* if and only if it is possible for the path flow to satisfy the bounds of both intervals. If the demand of all intermediate nodes of the path is zero, this is true if and only if they have nonempty intersection. Nonzero demands on path nodes can be reduced to this case by simple interval arithmetic which amounts to shifting intervals appropriately. An important observation is that the resulting (CPMC)-instance correctly models the problem, as a set of intervals is guaranteed to be compatible altogether if each pair of intervals is compatible – which is basically Helly's Theorem (see [Hel23]) in dimension 1. Finally, this instance is staircase, where (Rule 2a) follows from the fact that intervals are convex, and (Rule 2b) can easily be seen to hold from the way intervals can be sorted for each network edge.

Relation to General (CPMCS) The set of staircase instances that falls into one of the the two examples of (CPMCS) above forms a strict subclass of general staircase instances as defined in Definition 2.2. Intuitively, this is explained by the fact that many applications – including the two above – allow for some "transitivity reasoning", i.e. the compatibilities between subsets V_1 and V_2 together with those between V_2 and V_3 restrict the possible compatibilities between V_1 and V_3. However, according to the definition, both (Rule 2a) and (Rule 2b) only consider two subsets at a time. The

following example shows a compatibility graph that does not originate from either of the applications discussed above.

Example 2.5. *Consider the following compatibility graph G belonging to an instance of Problem (CPMCS) with three subsets $A = \{a_1, a_2, a_3\}$, $B = \{b_1, b_2, b_3\}$ and $C = \{c_1, c_2, c_3\}$, each of which has three elements. Note that there is another copy of subset A in the figure below to represent the compatibilities with subset C in order to highlight the symmetric structure of the example.*



Suppose G was obtained from an instance of Model (3)*. Then we could identify each subset with a task and each element of a subset with a possible starting time. We denote by $d_{AB} := \bar{d}_{AB} - \underline{d}_{AB}$ the length of the time window between tasks A and B and similarly for the other time windows. As $\{a_2, b_2\} \in E$, but $\{a_2, b_1\} \notin E$, $\{a_2, b_3\} \notin E$, we can conclude that d_{AB} is less than the time difference between b_1 and b_3, by slight abuse of notation denoted by $b_3 - b_1 > d_{AB}$. Due to c_2 being connected to all nodes in B, the time window of length d_{BC} has to include b_1 as well as b_3, and hence $d_{BC} \geq b_3 - b_1$, implying $d_{BC} > d_{AB}$. As the instance is symmetric, we can repeat this argument to obtain $d_{AC} > d_{BC}$ and $d_{AB} > d_{AC}$, which leads to the contradiction $d_{AB} < d_{BC} < d_{AC} < d_{AB}$.*

Similar reasoning shows that G also cannot be obtained from an instance of interval compatibilities on a path flow network (as it is described above): the argument is completely analogous, but uses the diameter of the intervals belonging to a_2, b_2, c_2 instead of d_{AB}, d_{BC}, d_{AC}.

Moreover, the situation is no different if we do not assume the ordering of the nodes within each subset as given. This is because there is no other ordering that makes $(G, \mathcal{V})$ a staircase instance (apart from simply reversing all subset orderings).

In Section 4, we will give computational results for the (CPMCS)-formulations derived in the following. The first class of instances will be from an application in railway timetabling which leads to a special project scheduling problem with precedence constraints. The second class of instances will come from a piecewise linearization of non-linear network flow problems, where we will use gas network optimization as an example.

3 Structural Properties

The problem (CPMC) introduced in the previous section can be modelled as a mixed-integer program (MIP) in a straightforward fashion. We introduce a variable $x_v \in \{0, 1\}$ for each node $v \in V$ which takes a value of 1 if this node is chosen and 0 if not. A vector x is then a feasible selection if and only if it is a solution to the following

feasibility problem:

$$
\begin{aligned}
\text{find} \quad & x \\
\text{s.t.} \quad & \sum_{v \in V_i} x_v = 1 && (\forall V_i \in \mathcal{V}) && \text{(4a)} \\
& x_v \leq \sum_{\substack{w \in V_j: \\ \{v,w\} \in E}} x_w && (\forall V_i \in \mathcal{V})(\forall v \in V_i)(\forall V_j \in \mathcal{V}, j > i) && \text{(4b)} \\
& x \in \{0,1\}^{|V|}. && && \text{(4c)}
\end{aligned}
$$

Multiple-Choice Constraints (4a) ensure that exactly one node from each subset in $\mathcal{V}$ is chosen, while Compatibility Constraints (4b) enforce the pairwise compatibility of the chosen elements according to the edge set E: choosing a node v from one subset V_i implies that we have to choose one of the nodes compatible to v in each of the remaining subsets V_j. Integrality Constraints (4c) finally restrict variables x to take binary values. Note that Constraints (4b) for two subsets V_i, V_j are redundant if $\{v, w\} \in E$ for all $v \in V_i$ and $w \in V_j$.

Remark 3.1. *It is easy to find examples where Constraints* (4c) *are actually needed as Constraints* (4a) *and* (4b) *are not sufficient to ensure integrality of the solution. For the instance presented in Example 2.4, Model* (4) *reads:*

$$
\begin{aligned}
\text{find} \quad & x \\
\text{s.t.} \quad & x_1 + x_2 + x_3 = 1 \\
& x_4 + x_5 + x_6 = 1 \\
& x_1 \leq x_4 + x_5 \\
& x_2 \leq x_5 + x_6 \\
& x_3 \leq x_5 + x_6 \\
& x \in \{0,1\}^6.
\end{aligned}
$$

It allows for the fractional solution $(0, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2})$ *if* $x \in \{0,1\}^6$ *is relaxed to* $x \geq 0$ *(observe that* $x \leq 1$ *is redundant). This solution is easily checked to be an extreme point of the corresponding polyhedron.*

Obviously, the polytope underlying Model (4) is not integral. However, we will see now that a small adaptation leads to a totally unimodular description of the feasible set, which is valid for staircase instances. We introduce the notation $\min(v, V_j)$ for some $v \in V_i \in \mathcal{V}$ and some V_j with $j \neq i$ to denote the minimal element in V_j which is compatible to v. Likewise, $\max(v, V_j)$ shall denote the maximal such element in V_j. Note that this relies on the fact that the V_i are ordered in a staircase instance. Consider then the feasibility problem given by

$$
\begin{aligned}
\text{find} \quad & x \\
\text{s.t.} \quad & \sum_{v \in V_i} x_v = 1 && (\forall V_i \in \mathcal{V}) && \text{(5a)} \\
& \sum_{\substack{u \in V_i: \\ u \geq v}} x_u \leq \sum_{\substack{w \in S_j: \\ w \geq \min(v, V_j)}} x_w && (\forall V_i \in \mathcal{V})(\forall v \in V_i)(\forall V_j \in \mathcal{V}, j \neq i) && \text{(5b)} \\
& x \in \{0,1\}^{|V|}. && && \text{(5c)}
\end{aligned}
$$

It uses the same set of variables as Model (4) as well as the same multiple-choice constraints to enforce (Rule 1). However, it features new compatibility constraints whose left-hand side arises by summing all x_u for $u \in V_i$ with $u > v$ onto the left-hand side of the old compatibility constraint corresponding to element $v \in V_i$ and some V_j with $j \neq i$. Its right-hand side arises by taking the old right-hand side and adding all variables x_w for $w \in V_j$ and $w > \max(v, V_j)$. Furthermore, this new model also incorporates compatibility constraints for subsets V_i and V_j with $i < j$. In the following, we show that the two models are in fact equivalent in the case of (CPMCS).

Proposition 3.2. *The respective feasible sets of Models* (4) *and* (5) *coincide if the compatibility graph G fulfils* (Rule 2a) *and* (Rule 2b).

Proof. We begin by showing that each feasible solution to Model (4) is also feasible for Model (5). To see this, consider a node v in a subset V_i and its corresponding compatibility constraint (4b) with the elements of another subset V_j, which reads

$$x_v \leq \sum_{\substack{w \in V_j: \\ \{v,w\} \in E}} x_w.$$

By summing up these constraints for all elements $u \geq v$, we obtain

$$\sum_{\substack{u \in V_i: \\ u \geq v}} x_u \leq \sum_{\substack{u \in V_i: \\ u \geq v}} \sum_{\substack{w \in V_j: \\ \{u,w\} \in E}} x_w = \sum_{w \in S_j} |\{u \in V_i \mid u \geq v, \{u,w\} \in E\}| \cdot x_w,$$

using (Rule 2b). Due to (Rule 2a), all coefficients for variable x_w with $\min(v, V_j) \leq w \leq w_{j,n_j}$ on the right-hand side of this inequality are exactly those which are non-zero, i.e. 1 or greater. As its left-hand side can at most take a value of 1 due to the multiple-choice constraint for subset V_i, all coefficients on the right-hand side greater than 1 can be reduced to 1 without changing the set of integer solutions fulfilling the inequality. This exactly yields Compatibility Constraints (5b), which proves that the feasible set of Model (4) is included in that of Model (5).

To prove the opposite inclusion, consider the case of a feasible solution to Model (5) which violates some compatibility constraint (4b) of Model (4). That would mean we have selected two nodes $v \in V_i$ and $w \in V_j$ with $\{v, w\} \notin E$. In other words, either $w < \min(v, V_j)$ or $w > \max(v, V_j)$ holds due to (Rule 2a). Assuming $i < j$, the first case can be ruled out, as a corresponding selection would violate Compatibility Constraint (5b) for node v and subset V_j. So let $w > \max(v, V_j)$. According to Compatibility Constraint (5b) for node w and subset V_i, we have $v > \min(w, V_i)$ and thus $v > \max(w, V_i)$ because of (Rule 2a). Now, as $\{v, \max(v, V_j)\} \in E$ and $\{w, \max(w, V_i)\} \in E$, we find $\{v, w\} \in E$ according to (Rule 2b), which is a contradiction. Consequently, we have shown that each feasible solution to Model (5) is feasible for Model (4). □

Note that similar as in the above proof, it can be shown that Compatibility Constraints (5b) for subsets V_i, V_j with $j < i$ are redundant to the corresponding constraints for $j > i$ if $\max(v, V_j) = w_{j,n_j}$.

Example 3.3. *Continuing the discussion of Example 2.4, we consider Model* (5) *for the associated problem instance, of which we already know that it is staircase:*

$$\begin{array}{lrcl}
\text{find} & x & & \\
\text{s.t.} & x_1 + x_2 + x_3 & = & 1 \\
& x_4 + x_5 + x_6 & = & 1 \\
& x_1 + x_2 + x_3 & \leq & x_4 + x_5 + x_6 \\
& x_2 + x_3 & \leq & x_5 + x_6 \\
& x_3 & \leq & x_5 + x_6 \\
& x_4 + x_5 + x_6 & \leq & x_1 + x_2 + x_3 \\
& x_5 + x_6 & \leq & x_1 + x_2 + x_3 \\
& x_6 & \leq & x_2 + x_3 \\
& x & \in & \{0,1\}^6.
\end{array}$$

This feasibility problem no longer allows for the fractional solution $(0, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2})$ *if relaxed to an LP.*

We will show next, that Model (5) is totally unimodular, which means that the corresponding polyhedron is integral.

Theorem 3.4. *The constraint matrix of Model* (5) *is totally unimodular.*

Proof. We use the following criterion to show that the constraint matrix corresponding to Model (5) is totally unimodular:

> A matrix A is totally unimodular, i.e. each square submatrix of A has determinant 0, $+1$ or -1, if and only if each collection of columns of A can be split into two parts, such that the sum of the columns in one part minus the sum of the columns in the other part is a vector with entries in $\{0, +1, -1\}$ only (see [GH62] or [Sch98, Theorem 19.3 (iv), p. 269]).

Given an arbitrary collection J of the columns of the constraint matrix, we partition J into subsets J_i according to the partition $\mathcal{V}$, where we assume w.l.o.g. that $J_i \neq \emptyset$ holds for all i. Then we compute the m column vectors that arise as the alternating sum of the columns in each J_i when going backwards through J_i (which exploits that the V_i are ordered) and giving the last column a positive sign. For each row belonging to a multiple-choice constraint, exactly one of the resulting m column vectors will have an entry of 1, the others an entry of 0. For the compatibility constraints recall that their supports are precisely the elements of two distinct subsets of $\mathcal{V}$. Thus, in a corresponding row, at most one column vector will have an entry $+1$, and at most one column vector will have an entry -1. The other entries will be 0. As a consequence, when summing the m column vectors, the result will be a column vector with entries in $\{0, -1, +1\}$ only. The submatrix $-I$ for the lower bounds need not be considered in the process, as they have no effect on total unimodularity. This concludes the proof. □

Example 3.5. *The constraint matrix from Example 3.3 has the following structure:*

$$\left(\begin{array}{ccc|ccc||c|c|c|c} \multicolumn{3}{c|}{V_1} & \multicolumn{3}{c||}{V_2} & \sum_{alt,V_1} & + & \sum_{alt,V_2} & \sum \\ \hline 1 & 1 & 1 & 0 & 0 & 0 & 1 & + & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & + & 1 & 1 \\ 1 & 1 & 1 & -1 & -1 & -1 & 1 & + & -1 & 0 \\ 0 & 1 & 1 & 0 & -1 & -1 & 0 & + & 0 & 0 \\ 0 & 0 & 1 & 0 & -1 & -1 & 1 & + & 0 & 1 \\ -1 & -1 & -1 & 1 & 1 & 1 & -1 & + & 1 & 0 \\ -1 & -1 & -1 & 0 & 1 & 1 & -1 & + & 0 & -1 \\ 0 & -1 & -1 & 0 & 0 & 1 & 0 & + & 1 & 1 \end{array}\right),$$

where we have left out the variable bounds. For the sake of simplicity, we consider the case $J = \{1, \ldots, 6\}$ *from the proof of Theorem 3.4. When computing the alternating sum* $\sum_{alt,V_1}$ *of the columns corresponding to the elements of subset* V_1*, going backwards and starting with a positive sign in the last column, we observe that this yields a column vector that only consists of entries in* $\{0, +1, -1\}$*. This also holds for the alternating sum* $\sum_{alt,V_2}$ *of the columns corresponding to the elements of subset* V_2 *computed in the same fashion. For the rows corresponding to Multiple-Choice Constraints* (5a)*, exactly one of the two column vectors contains an entry* $+1$ *and the other one an entry* 0*. For the rows corresponding to Compatibility Constraints* (5b) *for the elements of* V_1*, the* V_1*-column vector contains either a* $+1$ *or a* 0 *and the* V_2*-column vector either a* -1 *or a* 0*, and vice versa for the elements of* V_2*. Thus, when adding the two column vectors, the result is a new column vector whose entries are in* $\{0, -1, +1\}$ *only. This property still holds when forming a submatrix by deleting individual columns of the constraint matrix due to the staircase structures in the compatibility constraints. Therefore, we have shown the total unimodularity of the matrix and this way also the integrality of the polyhedron.*

In many cases, totally unimodular constraint matrices correspond to problems defined on a network. More precisely, the matroid formed by a totally unimodular constraint matrix can be decomposed into matroids that are graphic, cographic or isomorphic to the special matroid R_{10} (on the decomposition of regular matroids, see [Sey80]) – where the latter is neither graphic nor cographic and rarely occurs in practical applications. Thus, it is natural to ask the question whether the constraint matrix of Model (5) is graphic or cographic (i.e. the linear matroid obtained from the matrix is a graphic or cographic matroid), in which case (CPMCS) is equivalent to a network flow problem or a dual network flow problem ("potential problem") respectively. The reader not familiar with these notions of matroid theory may consult [Oxl06].

Theorem 3.6. *The constraint matrix of Model* (5) *is cographic.*

Proof. We show this by transforming Model (5) into a dual network flow problem. Given a directed graph $G = (V, A)$, such a problem is of the form

$$\begin{aligned} \text{find} \quad & y \\ \text{s.t.} \quad & y_v - y_u \leq d_{uv} \qquad (\forall a = (u, v) \in A) && \text{(6a)} \\ & y \in \mathbb{R}^{|V|}, && \text{(6b)} \end{aligned}$$

where $d \in \mathbb{R}^{|A|}$ and y is an unrestricted continuous variable. To obtain this form, we use the following variable transformation: let

$$y_{ij} := \sum_{k=j}^{n_i} x_{ik} \quad (\forall i = 1, \ldots, m)(\forall j = 1, \ldots, n_i),$$

where x_{ik} denotes the x-variable from Model (5) corresponding to the i-th vertex in subset V_k. Note that this transformation is bijective with $x_{ij} = y_{ij} - y_{i,j+1}$ if $j < n_i$, and $x_{i,n_i} = y_{i,n_i}$. Stating Model (5) in terms of the y-variables, we see that both sides form telescope sums, leaving only one variable on each side. Thus, the compatibility constraint (5b) for two subsets V_i and V_l and some $v \in V_i$ now reads

$$y_v - y_{\min(v,V_l)} \leq 0,$$

which has the form of (6a). The multiple-choice constraints modelling (Rule 1) translate into

$$y_{i,1} = 1 \quad (\forall i = 1, \ldots, m). \tag{7}$$

This also implies upper bounds on the x-variables. Their lower bounds can be expressed via

$$y_{i,j+1} - y_{ij} \leq 0 \quad \text{if } j < n_i, \quad \text{and } -y_{i,n_i} \leq 0 \quad (\forall i = 1, \ldots, m). \tag{8}$$

The graph G is then simply defined to have a node for every y-variable and an arc (u, v) if and only if there is a constraint $y_v - y_u \leq 0$. □

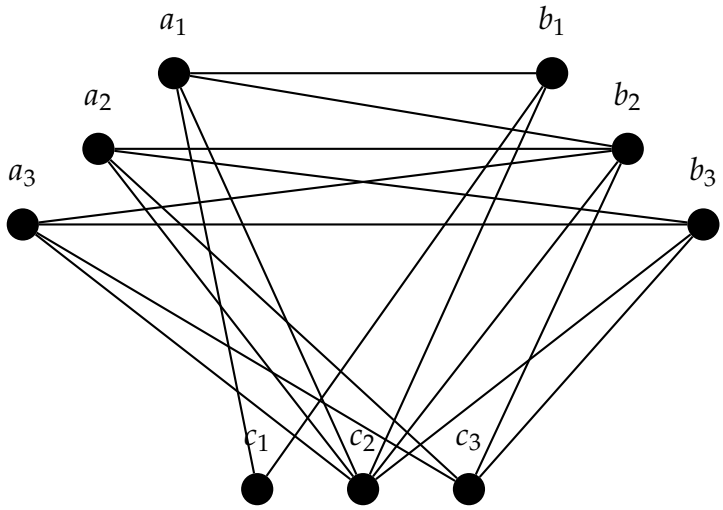
Using the transformation from the above proof, we obtain the following dual-flow formulation for (CPMCS):

$$\begin{aligned}
&\text{find} && y && && \\
&\text{s.t.} && y_{i,1} &= 1 && (\forall V_i \in \mathcal{V}) && \text{(9a)}\\
& && y_v &\leq y_{\min(v,V_j)} && (\forall V_i \in \mathcal{V})(\forall v \in V_i)(\forall V_j \in \mathcal{V}, j \neq i) && \text{(9b)}\\
& && y_{i,j+1} &\leq y_{ij} && (\forall V_i \in \mathcal{V})(\forall j < n_i) && \text{(9c)}\\
& && y_{i,n_i} &\geq 0 && (\forall V_i \in \mathcal{V}). && \text{(9d)}
\end{aligned}$$

Remark 3.7. *The y-variables have the following interpretation: $y_{ij} = 1$ means: "from V_i, pick an element with index j or greater". This is closely related to the* incremental method *for linearizing a univariate function. Furthermore, the above transformation is well known in this context (see [Vie15]), where it is used to connect the incremental method to, for example, the* convex-combination method, *and vice versa. We can also recognize* (8) *as a* filling condition.

The following example illustrates the transformation of (CPMCS) to a dual network flow problem. It will also show that the constraint matrix is not graphic in general.

Example 3.8. *Let V be partitioned into three subsets $A = \{a_1, a_2, a_3\}$, $B = \{b_1, b_2, b_3\}$ and $C = \{c_1, c_2, c_3\}$. Let $G = (V, E)$ then be the following compatibility graph, where each pair of subsets behaves as in Example 2.4:*



As described in the proof of Theorem 3.6, Compatibility Constraints (5b) *turn into Inequalities* (3)*, e.g. considering node a_2 together with subset B, the corresponding inequality*

$$x_{a_2} + x_{a_3} \leq x_{b_2} + x_{b_3}$$

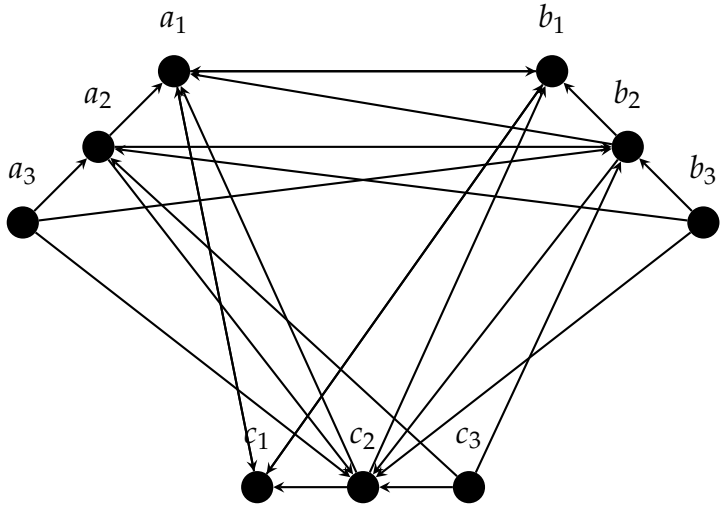
in terms of the y-variables now reads

$$y_{a_2} \leq y_{b_2}.$$

More generally, using the notation from the proof of Proposition 3.2, for each node $v \in V_i$ and each subset V_j, $j \neq i$, we have

$$y_s \leq y_{\min(v,V_j)}.$$

Due to (8)*, there are additional constraints ordering the y-variables within each subset. Therefore, by the proof of Theorem 3.6 we can formulate the given instance of (CPMCS) as a dual network flow problem on the following directed graph (arcs may be read as implications).*



The example shows that the constraint matrix of Model (5) *is not graphic in general, as this would require the above graph to be planar. However, this is not the case, as, for example, it has $K_{3,3}$ as a subgraph, induced by nodes $\{a_2, b_2, c_2\}$ and $\{a_3, b_3, c_3\}$.*

As the constraint matrix is totally unimodular, we are guaranteed that each fractional point is the convex combination of integral solutions. Next, we will show how to

find such a convex combination. In the case where (CPMCS) forms a substructure of a more complex problem, this may be useful for constructing a heuristic, as the integer points spanning a fractional solution are candidates for good feasible solutions.

The following is an easy way to obtain integral solutions from a fractional one. It generalizes the well-known fact that for dual network flow problems, rounding all components up or rounding all component down preserves feasibility.

Definition 3.9. *Let $\hat{y}$ be a solution to Model* (9) *for (CPMCS), and let $\lambda \in (0,1]$ be some threshold value. We define $\hat{y}^\lambda$ to denote the integer point obtained from rounding all components of $\hat{y}$ according to the following rule:*

$$\hat{y}_i^\lambda = \begin{cases} 1 & \text{if } \hat{y}_i \geq \lambda \\ 0 & \text{if } \hat{y}_i < \lambda \end{cases},$$

*and say that $\hat{y}^\lambda$ is obtained from λ-*rounding *$\hat{y}$.*

It is easy to see that $\hat{y}^\lambda$ is also a solution to Model (9), for all $\lambda \in (0,1]$. The key observation is that every operation that does not change the relative ordering of the y_{ij} (and also does not violate the 0/1-bounds), preserves feasibility, as $d_{ij} = 0$ in (6a) whenever there are two variables present in the constraint.

Theorem 3.10. *Let $\hat{y}$ be a solution to Model* (9)*. Then $\hat{y}$ is a convex combination of the integral solutions*

$$\{\hat{y}^\lambda \mid \lambda \in \{\hat{y}_i,\, i = 1, \ldots, |V|\}\}.$$

Proof. Let $\Lambda := \{\hat{y}_i,\, i = 1, \ldots, |V|\}$ denote the set of values occurring in $\hat{y}$. We denote them by $\lambda_1, \ldots, \lambda_{|V|}$ and assume they are ordered increasingly, i.e. $\lambda_i \leq \lambda_j$ whenever $i \leq j$. We claim that

$$\hat{y} = \sum_{k=1}^{|V|} (\lambda_k - \lambda_{k-1}) \hat{y}^{\lambda_k}, \tag{10}$$

where λ_0 is interpreted as 0. Indeed, the i-th component of $\sum_{k=1}^{|V|} (\lambda_k - \lambda_{k-1}) \hat{y}^{\lambda_k}$ is equal to

$$\begin{aligned} \sum_{k=1}^{|V|} (\lambda_k - \lambda_{k-1}) \hat{y}_i^{\lambda_k} &\overset{\text{Def. 3.9}}{=} \sum_{k:\lambda_k \leq \hat{y}_i} (\lambda_k - \lambda_{k-1}) 1 \\ &\overset{\text{telescope sum}}{=} \max_{k:\lambda_k \leq \hat{y}_i} \lambda_k - \underbrace{\lambda_0}_{=0} \\ &= \hat{y}_i. \end{aligned}$$

Furthermore, we have

$$\lambda_k - \lambda_{k-1} \geq 0 \text{ for all } k = 1, \ldots, |V|,$$

and also

$$\sum_{k=1}^{|S|} (\lambda_k - \lambda_{k-1}) = \lambda_{|V|} - \lambda_0 = 1,$$

since (7) implies $1 \in \Lambda$, and therefore $\lambda_{|V|} = 1$. Thus, Equation (10) describes $\hat{y}$ as a convex combination of $\{\hat{y}^\lambda \mid \lambda \in \Lambda\}$. □

4 Computational Results

In this section, we compare the efficiency of the three MIP formulations for (CPMCS) we have discussed previously: the first, naive compatibility formulation (4), the totally unimodular compatibility formulation (5) and the formulation (9) as a dual network flow problem. We do this by evaluating them on real-world benchmark instances arising from two different applications: energy-efficient railway timetabling and piecewise linearization of the physical flow constraints on gas networks. We will see that passing from the original to the unimodular formulation already brings a significant computational advantage, but that the much sparser dual-flow formulation allows for the best results by far. Our computational study thus demonstrates two more things: staircase structures are present in real-world application problems, and their exploitation is very beneficial in terms of computation time.

4.1 Computational Results for Energy-Efficient Timetabling

The first example for a successful exploitation of staircase compatibility we present here is a problem in railway timetabling. The aim is to take a timetable draft which is currently in the planning phase (typically towards the end) and to use the remaining degrees of freedom to allow for a reduction of the energy costs of the involved train operating companies (TOCs). This is possible by taking into account that a big consumer of electricity (as a TOC undoubtedly is) typically has an electricity contract consisting of two price components: the overall energy consumption and the maximum average power consumption over all 15-minute intervals in the billing period. In the special case of a German TOC, the electricity provider charges the collective energy and power consumptions of all the trains operated by this TOC. This is done by summing up their individual power consumption profiles as measured by the electricity meters in the locomotives and computing both the area under the resulting curve (i.e. the total energy consumption) as well as the maximum 15-minute average. Both values are multiplied with some respective cost factor and summed to obtain the final electricity bill. One possibility for optimization via timetabling now lies in adjusting the departure times of the trains in the stations. A train generally draws most power while accelerating. Thus, high peaks in power consumption can be avoided if too many simultaneous departures are desynchronized, which can be used to decrease the price component based on peak power consumption. In many cases, this effect can already be achieved via small shifts in the departure times and is thus an interesting trade-off to be considered: the power-based price component typically makes up for 20–25% of the energy bill.

We illustrate the effect of this optimization in Figure 1. It shows the power consumption profile before and after optimization for one of the benchmark instances introduced later (*Würzburg*) on a sample day. The curves in *red* show the power consumption in each second, while the *blue* curves show their consecutive 15-minute averages. As stated, the TOC is charged proportionally to the highest such average over the billing period (typically one year). According to the official price sheet by DB Energie GmbH for 2017, the cost factor is 125.94 € per kW and year, such that the demonstrated reduction from 79 to 72 MW in peak power consumption equals an annual cost saving of around 880,000 € (and this is a rather small instance). Note that the energy recuperated from braking trains is refunded separately and not offset against the power drawn

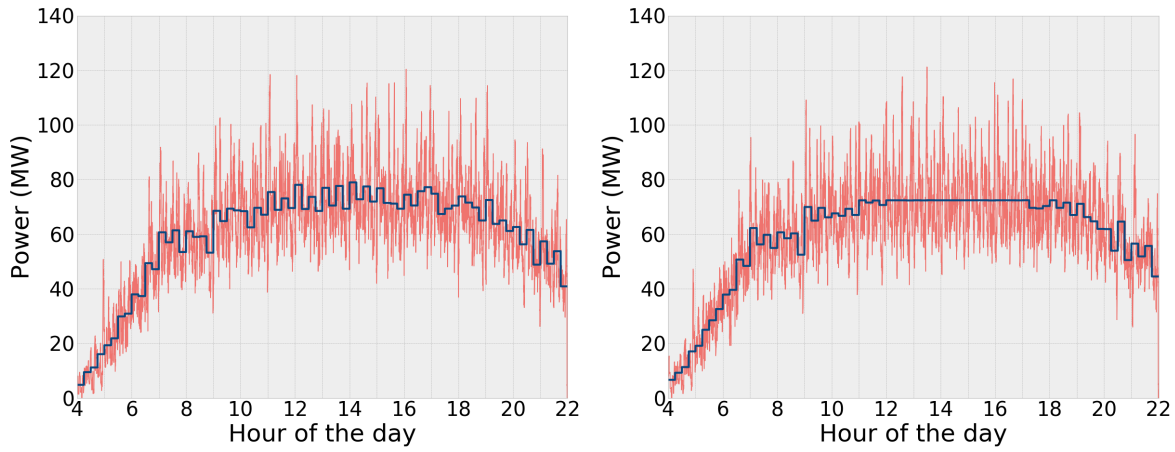


Figure 1: Power consumption profile of timetabling instance *Würzburg* before (left) and after (right) optimization

from the power supply. Thus, we can assume that the power consumption profile is always non-negative.

In the following, we give a statement of the problem in terms of staircase compatibility and present our computational study on problem instances of different sizes.

The Problem as a Special Case of (CPMCS) We consider a given initial timetable in which each departure time of a train from a station may be shifted within some interval around the current departure time. We assume that the travel times of the trains on the tracks as well as the corresponding power consumptions are fixed. Furthermore, we assume that the temporal order of the trains passing a certain track may not be changed by the optimization and that all connections between different trains in a station must be preserved in order to retain the structure of the original timetable as far as possible. Assuming a fixed order of the trains on each track, we also know the safety distances to respect between each consecutive pair of trains. The problem is now to find an adjusted timetable that minimizes the maximum average power consumption.

In order to state this problem in terms of (CPMCS), we need to define the node set V and the edge set E of the corresponding compatibility graph G. Let the set T be the planning horizon with a resolution in seconds, let R be the set of all trains and let A^r be the set of legs that train $r \in R$ travels. A trip of a given train between two stations visited in succession is what we call a *leg*. We denote by $J^a \subseteq T$ the set of all feasible departure times for a leg $a \in A := \bigcup_{r \in R} A^r$ within the planning horizon T. The travel time for a train $r \in R$ on a leg $a \in A^r$ is Γ^a.

We choose V to be the set of all pairs (a, j) of a leg $a \in A$ and its feasible departure times $j \in J^a$. It is then natural to choose the partition $\mathcal{V} = \bigcup_{a \in A} V_a$, where V^a is the set of all feasible pairs (a, j) for some fixed $a \in A$. A timetable adaptation is then made up of a selection of exactly one node from each subset V^a. To be feasible, this selection has to respect several further constraints which are stated in the following. Firstly, these are the dwell time constraints: upon completing a leg $a_1 \in A^r$ and departing for the subsequent leg $a_2 \in A^r$, the train has to stop for a minimum time of $c^{a_1 a_2}$. Let W denote the set of all such pairs of successive legs of the same train. Secondly, we have

the minimum headway time constraints: for each pair of legs (a_1, a_2) corresponding to consecutive trains (r_1, r_2) passing the same track, as given by a set L, we have to keep to a minimum headway time of $s^{a_1 a_2}$. We assume that it is sufficient with respect to safety to ensure that the respective times of departure and arrival of these two legs are separated by this minimum headway time. Finally, we consider the connection constraints. Let r_1, r_2 be two trains which meet in a given station and for which changing from r_1 to r_2 shall be possible. Let a_1 be the leg that corresponds to the arrival of r_1 in that station and a_2 the leg that corresponds to the departure of r_2 from that station. Then the time that passes between the arrival of r_1 and the departure of r_2 must be at least $\rho^{a_1 a_2}$ and at most $\theta^{a_1 a_2}$. Let the set U contain all pairs of legs with this interchange requirement.

The edge set E stating the compatibility between any two nodes $(a_1, j_1), (a_2, j_2) \in V$ is now given by

$$E := E_1 \cap E_2 \cap E_3.$$

Here, the set of edges E_1 models the compatibility according to the minimum dwell times:

$$E_1 := \{\{(a_1, j_1), (a_2, j_2)\} \mid ((a_1, a_2) \in W, j_2 \geq j_1 + \Gamma^a + c^{a_1 a_2}) \\ \vee ((a_1, a_2) \in (A \times A) \setminus W, a_1 \neq a_2)\},$$

edge set E_2 models the compatibility according to the minimum headway times:

$$E_2 := \{\{(a_1, j_1), (a_2, j_2)\} \mid ((a_1, a_2) \in L, j_2 \geq j_1 + \max(0, \Gamma^{a_1} - \Gamma^{a_2}) + s^{a_1 a_2}) \\ \vee ((a_1, a_2) \in (A \times A) \setminus L, a_1 \neq a_2)\},$$

and edge set E_3 models the compatibility according to the connection times:

$$E_3 := \{\{(a_1, j_1), (a_2, j_2)\} \mid ((a_1, a_2) \in U, j_2 \geq j_1 + \Gamma^{a_1} + \rho^{a_1 a_2} \wedge j_2 \leq j_1 + \Gamma^{a_1} + \theta^{a_1 a_2}) \\ \vee ((a_1, a_2) \in (A \times A) \setminus U, a_1 \neq a_2)\}.$$

It is easy to check that for each $G_i = (V, E_i)$, $i = 1, 2, 3$, $(G_i, \mathcal{V})$ is a staircase instance. Likewise, it is easy to check that intersecting the edge sets of any number of staircase instances on the same node set and with the same partition preserves this property. Consequently, $G = (V, E)$ with $E := E_1 \cap E_2 \cap E_3$ is a staircase instance, which allows us to formulate the set of feasible selections according to each of the three models derived in Section 3.

For each leg $a \in A$ and each allowable departure time $j \in J^a$, we define a variable $x_j^a \in \{0, 1\}$, which takes a value of 1 if leg a departs at time j, and 0 otherwise. Furthermore, we introduce the set X of all feasible timetables, which can be modelled via the naive or the totally unimodular compatibility formulation of (CPMCS). Accordingly, we write $x \in X$ to denote the constraints modelling a feasible timetable in short. What is then left to define is the objective function. Let $p^{at} \geq 0$ be the power consumption of a train r when travelling leg $a \in A^r$ at point $0 \leq t \leq \Gamma^a$ after departure. Consequently, if train r departs for leg a at time j, the power consumption at time $t \in T$ is given by:

$$\bar{p}_j^{at} := \begin{cases} p^{at}, & \text{if } 0 \leq t - j \leq \Gamma^a \\ 0, & \text{otherwise.} \end{cases}$$

Let $I := \{1, 2, \ldots, m\}$ be the set of the m consecutive 15-minute (= 900-second) intervals in T (where the last interval may actually be somewhat shorter). The total energy

consumption of train r for travelling leg a that falls into the averaging interval $i \in I$ when choosing departure time $j \in J^a$ is then given by

$$e_j^{ai} := \frac{1}{2}(\bar{p}_j^{a,900i} + \bar{p}_j^{da,900(i+1)}) + \sum_{t=900i+1}^{900(i+1)-1} \bar{p}_j^{at}$$

(we consider the power consumption p as a piecewise-linear function over time). The average power consumption over an interval $i \in I$ by all trains depending on the chosen departure times is then given by

$$z_i(x) := \frac{1}{900} \sum_{a \in A} \sum_{j \in J^a} e_j^{ai} x_j^a.$$

This leads to the following optimization problem to minimize the highest of these averages:

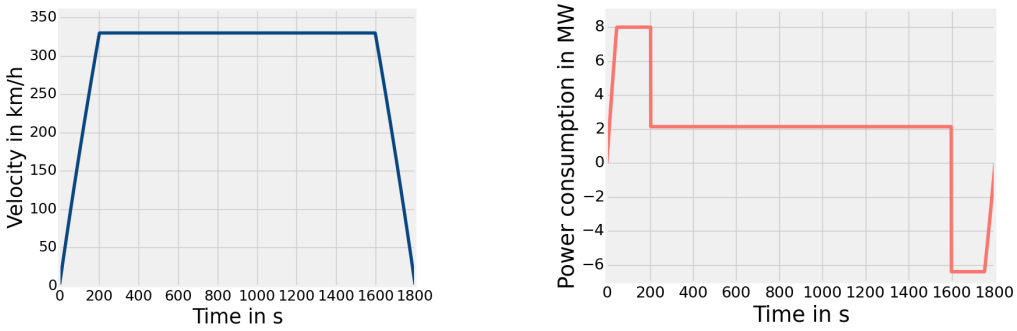
$$\min_{x \in X} \max_{i \in I} z_i(x), \tag{11}$$

Via the transformation from Theorem (3.6) from Section 3, it is straightforward to derive the corresponding dual-flow formulation for the above timetabling problem. Note that Problem (11) is NP-hard even if all trains only have one leg and $m = 2$, as can easily be shown by a reduction from the partition problem (see [GJ79, SP12]). This is no contradiction to the total unimodularity of the (CPMCS)-formulation for the feasible timetables: we are not optimizing over a linear objective function here, but over a piecewise-linear convex objective function.

Computational Comparison of the Models for (CPMCS) We now present a computational study that compares the different formulations for staircase compatibility considered before as a part of the timetabling problem introduced above. We do this on real-world instances derived from the 2015 timetable for the German passenger traffic operated by our industry partner Deutsche Bahn AG (DB). We complemented this data by power consumption profiles based on height data of the stations as well as simplified velocity profiles taking into account train characteristics. An example is depicted in Figure 2 which shows an assumed velocity profile for an ICE-3 on a journey of 30 minutes in Figure 2a and the corresponding power profile on a track with an upwards inclination in Figure 2b. The minimum headway times we chose are based on [Pac16, Table 5.4] by rounding up the given values to full minutes.

Altogether, we have created 31 instances of different sizes, each for a planning horizon of 18 hours (4 a.m. to 10 p.m.). These contain 18 *local instances* which contain all trains passing a certain station in Germany, 10 *regional instances* which contain all short-distance trains circulating in a given region of Germany, 1 *Regionalverkehr instance* comprising the traffic of all regional instances together, 1 *Fernverkehr instance* covering the German long-distance traffic, as well 1 *Deutschland instance* including all German DB passenger trains. Each instance contains those parts of the journeys of the involved trains which fall within the planning horizon. The allowable shift in departure time was uniformly chosen to be ± 3 minutes around the current departure time.

The computations have been carried out on a queuing cluster of Intel Xeon E5-2690 3.00 GHz computers with 25 MB cache and 128 GB RAM, using 5 threads and a time limit of 10 hours. Our implementation uses the Python-API of *Gurobi 7.5.2* (see [Gur17]).



(a) Velocity profile

(b) Power consumption profile

Figure 2: Example profiles for an ICE-3 on a 30 minutes journey climbing an inclination

The sizes of the created instances, the computation times of the three different models as well as the achieved savings in peak power consumption are shown in Table 1. Here, NA denotes the naive formulation (4), TU stands for the totally unimodular formulation (5), and DF represents the formulation as a dual network flow problem. The result is very clear: Formulation DF is by far the best way to formulate the compatibilities as it leads to the fastest solution times on all but one instance. In many cases, the benefit is very significant. Most notably, for the Germany-wide instance *Deutschland* the computation time could be decreased from about 3 hours to little more than 3 minutes – a factor of more than 50. The table also shows that the computation time of Formulation TU is usually between the solution times required for Formulations NA and DF. This shows the general benefit of passing to a totally unimodular description of the set of feasible timetables. However, the sparsity of Formulation DF is what most likely leads to the much lower node solution times in the branch-and-bound tree we observed and is therefore vastly superior. We remark here that the stated reduction of about 5% in peak power consumption for the Germany-wide instance would allow for cost savings of several million euros per year. More detailed information on the problem can be found in [BMS17].

4.2 Computational Results for Piecewise Linearization of Path Flows

Our second example for a staircase compatibility structure originates from piecewise linearization of a non-linear network flow problem. In order to deal with non-linearities, a common and successful approach consists in constructing piecewise linearizations or relaxations of the involved non-linear functions (see Figure 3), which makes the problem accessible to general-purpose MIP solvers. It involves subdividing the feasible set into several intervals and introducing binary variables that indicate which interval the argument value is in. For a detailed description of the this technique, see [GMMS12].

For the purpose of this section, we do not have to consider the non-linearity explicitly, but just assume that it can be modelled as a univariate function of the arc flow. As an example for such a type of problem, we consider gas network optimization. Here, in addition to a classical network flow problem for the mass flow, pressure has to be considered as well. Therefore, pressure variables are introduced for network nodes, and additional constraints describe the non-linear pressure loss along pipes. A commonly-used algebraic approximation of the underlying physics is given in [PFG$^+$15, Equa-

Instance	#Trains	#Trips	Computation time [s] NA	TU	DF	Saving [%]
Zeil	42	762	5.92	3.21	**1.08**	14.89
Bayreuth Hbf	68	327	0.44	0.50	**0.24**	22.18
Passau	75	1040	366.46	311.53	**15.69**	14.48
Jena Paradies	78	1102	154.21	26.99	**11.16**	12.46
Lichtenfels	113	1650	695.77	555.71	**11.69**	15.25
Erlangen	142	2969	3326.92	516.13	**39.50**	15.28
Bamberg	209	3644	8676.92	131.64	**16.23**	13.08
Aschaffenburg	245	3463	45.76	13.40	**2.05**	12.95
Kiel Hbf	297	2130	128.67	27.47	**3.28**	11.19
Leipzig Hbf (tief)	369	6810	8.65	4.44	**0.71**	6.40
Würzburg Hbf	371	4456	35408.58	19303.35	**1470.83**	8.32
Dresden	422	6936	35701.15	703.83	**53.18**	9.29
Ulm Hbf	468	5729	2930.31	50.93	**7.71**	11.23
Stuttgart Hbf (tief)	628	11594	910.35	504.93	**11.87**	0.93
Berlin Hbf (S-Bahn)	639	16114	21.11	30.79	**13.99**	2.97
Hamburg-Altona(S)	722	12373	689.98	151.77	**10.43**	1.29
Frankfurt(Main)Hbf	728	8626	5033.25	658.46	**34.40**	10.28
Nürnberg	951	12189	16484.25	46.95	**6.13**	7.10
S-Bahn Hamburg	1208	17533	1497.44	338.08	**19.37**	2.36
Regio Nord	1476	13379	94.39	22.13	**8.56**	12.79
Regio Nordost	1494	16496	1304.01	66.47	**11.28**	15.50
Regio Hessen	1547	25092	32.92	34.75	**11.61**	5.64
Regio Südwest	1863	24191	203.87	66.22	**9.94**	13.00
Regio Südost	2357	31917	409.74	142.94	**19.45**	8.95
Regio BW	2382	30172	2629.29	466.83	**24.51**	13.35
S-Bahn Berlin	2578	53353	**89.02**	250.26	176.05	1.73
Regio NRW	2826	47026	1497.60	825.87	**32.54**	5.13
Regio Bayern	3554	49262	764.46	114.13	**51.08**	10.72
Fernverkehr	667	7053	289.23	16.98	**11.81**	5.38
Regionalverkehr	21288	308472	27233.03	5709.24	**4548.82**	9.53
Deutschland	21955	315525	10569.42	3948.35	**197.28**	5.05

Table 1: Computational results for the three problem formulations for the energy-efficient timetabling problem

tion (7)]. For a constant compressibility factor (see [PFG+15, Equation (20)]), the reduction in squared pressure Δp^2 on a pipe is indeed a non-linear univariate function of the flow q on that pipe, given by

$$\Delta p^2 = \lambda q|q|$$

for some $\lambda > 0$.

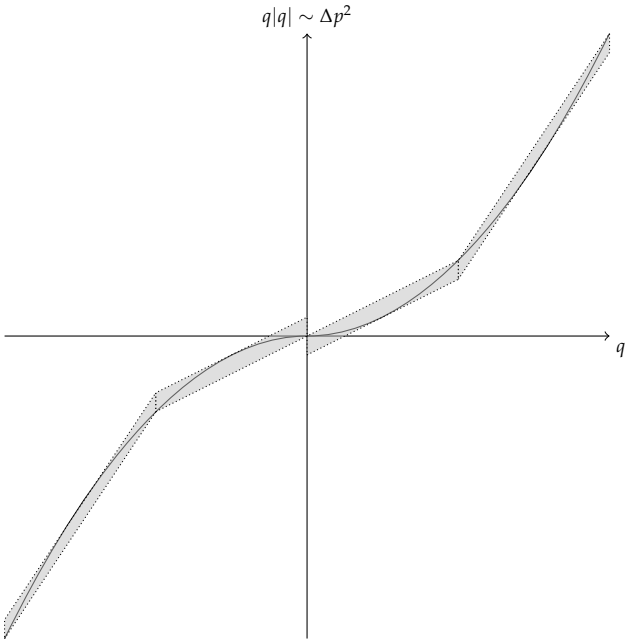


Figure 3: Illustration of a piecewise-linear relaxation (shaded parallelograms) of a univariate non-linear function

It should be mentioned that real-world gas network optimization problems usually contain additional challenges connected with the operation of active elements such as valves and compressors. However, as the substructure we are interested in can be motivated just from passive networks, we do not want to go into detail here and instead refer the reader to the literature, e.g. [KHPS15]. As a proof of concept, we will consider random instances using a real-world gas network topology for the computational experiments below.

For constructing a piecewise-linear approximation, i.e. for modelling a piecewise-linear function, several useful formulation methods are known. For now, we assume that we use a method where there is a binary variable z_I for each interval $I = [l_I, u_I]$ with the meaning

$$z_I = 1 \quad \Rightarrow \quad q_a \in [l_I, u_I], \tag{12}$$

as is true, for example, for the *Multiple-Choice Method* (MCM, [JL84]) as well as for the *Convex-Combination Method* (CCM). For a single piecewise-linear function, MCM leads to *locally ideal* formulations, i.e. their linear relaxation is equal to the convex hull of feasible points. The basic version of CCM is not locally ideal; however, a locally ideal improved variant is proposed in [Pad00]. However, the situation is different when we consider multiple non-linear functions that influence each other: in general, the formulation loses its desired property of being ideal.

For a network that is a path of length k, we can regain a complete description in the context of staircase compatibility. This is also covered as a special case in [LM16], though using different reasoning. We have already seen in Subsection 2.1 that this represents a special case of (CPMCS).

Computational Comparison of the Models for (CPMCS) We will also test the impact the different formulations for staircase compatibility have on instances arising from piecewise-linearized network flow problems. We use the following setting for our test instances: given a network, for all arcs a we have real-valued flow variables q_a that have to satisfy flow conservation and demand satisfaction equations

$$\sum_{a \in \delta^+(v)} q_a - \sum_{a \in \delta^-(v)} q_a = d_v$$

for all network nodes, where d_v denotes the given demand for node v.

Furthermore, for each arc a we have several possible intervals $I_{a,1}, \ldots, I_{a,n_a}$ for the arc flow, where n_a denotes the number of intervals belonging to arc a. Intervals belonging to the same arc arise from subdividing a larger interval, such that they intersect in at most a point and come with a natural ordering. Let z_{aj} denote binary indicator variables for using the j-th interval on arc a. Lower and upper bounds of an interval I_{aj} are denoted by l_{aj} and u_{aj} respectively. Only one interval per arc can be active – and at least one has to. Therefore, the corresponding z-variables are connected by the constraint

$$\sum_{j=1}^{n_a} z_{aj} = 1,$$

which represents (Rule 1).

For implementing (12), the multiple-choice method is used. Recall that a piecewise-linear approximation of a univariate function f of q modelled by MCM on a connected domain $D = [l_D, u_D]$ with breakpoints $B_1 = l_D, B_2, \ldots, B_k, B_{k+1} = u_D$ is obtained for $l_{a_i} = B_i, u_{a_i} = B_{i+1}$, $i = 1, \ldots, k$. We create a "copy" q_i of the arc flow q for every interval and ensure (12) by the constraints

$$l_i z_i \leq q_i \leq u_i z_i \quad (\forall i = 1, \ldots, k),$$

where arc indices of the z-variables are omitted for better readability, as we only consider a single arc. Then, q and $f(q)$ can be expressed as follows:

$$q = \sum_{i=1}^{k} q_i, \quad f(q) = \sum_{i=1}^{k} f(B_i) z_i + (q_i - B_i z_i) \frac{f(B_{i+1}) - f(B_i)}{B_{i+1} - B_i}.$$

These constraints constitute our polyhedron of interest. The underlying network is given by the topology of a real-world gas network by the German gas network operator *Open Grid Europe* (OGE) consisting of 592 nodes and 623 arcs. As the network is not a path, there is no complete description available. However, (CPMCS) is present as a substructure, e.g. at each induced path of degree-2 nodes in the network. 224 nodes have degree two and there are 128 paths of degree-2 nodes, which amounts to an average length of 2.75. The longest of these paths has length 8. In the following, we want to test the effect of using our improved formulations of (CPMCS) in these places. Note, however, that these formulations may also be applied to substructures that form

induced paths of degree-2 nodes if we only consider edges for which the flow is not fixed. This includes structures in which middle nodes might have junctions but the flow contribution from outside the substructure can be fixed during preprocessing.

We first identify all suitable subpaths of degree-2 nodes in the network, construct the corresponding compatibility graphs and precompute the unimodular formulation of Model (5) for each of the detected subpaths. This description is quadratic in the length of the path and linear in the number of intervals per arc. We do not add all these constraints right from the start as in practice many of them are redundant. Instead, we use a separation callback at every 50th branch-and-bound node that finds all violated inequalities and adds them to the model. The callback is called at most 100 times, after which the most relevant cutting planes typically have been added already.

For all test instances additional input data is generated at random. This includes the vector d of demands as well as the initial arc capacities c. Capacities were scaled in such a way that feasibility of all instances is guaranteed. We then chose a random partition of the interval $[-c_a, c_a]$ into a given constant number of intervals for each network arc. This number varies across different instance sets and is meant to roughly represent the accuracy of the linearization. The objective function is constructed by drawing integer coefficients for the z-variables. This is done uniformly at random from the interval from 0 to twice the number of intervals per arc, with the restriction that there is an upper bound on the resulting "slope" of the objective function. As the generation of instances includes randomness, we always generated sets of five instances with the same number of intervals per arc. The solution times given in the following are always (geometric) averages over five instances each. If only a subset of the five instances was solvable within the time and memory limitations, the average is taken over this subset only. Note that for ranking the methods, top priority is given to the number of instances solved. If a method did not solve any of the five instances of a given set, we denote this by an average solution time of ∞.

The computational experiments have been performed on a queuing cluster of Intel Xeon E5-2690 3.00 GHz computers with 25 MB cache and 128 GB RAM. Our implementation uses the C++-API of *Gurobi 6.0.0*. For large numbers of intervals per arc, we encountered numerical difficulties on the test set. These numerical issues are already observed in the standard formulation. To overcome this, we increased Gurobi's parameter *NumericFocus* to the value of 3 in order to tell the solver to be more careful regarding numerical issues. As a result, we did not observe numerical difficulties for any of the instances any more. However, this choice results in longer running times. Apart from that, we use Gurobi's standard parameter settings, except for turning on *PreCrush* for our cutting plane methods, which is mandatory if we want to add user cuts. Each job was run on 4 cores and with a time limit of 40 hours of CPU-time.

As can be seen from Table 2, adding constraints from the totally unimodular formulation of the (CPMCS)-substructures (i.e. paths of degree-2 nodes) improves the runtime of the solver considerably for most test sets. This effect increases with a growing number of intervals per arc, resulting in a total of 2 more instances that can be solved within the time limit.

Next, we want to apply the dual-flow formulation (see Model (9)). Note, however, that the transformation used to obtain this formulation and to prove Theorem 3.6 is well known in the present context for connecting the incremental method to methods using (12), as already mentioned in Remark 3.7.

Let us briefly recall the incremental method (or δ-method), first introduced in [MM57]:

# intervals per arc	MCM solved	MCM CPU[s]	MCM + TU-paths solved	MCM + TU-paths CPU[s]
3	**5**	**66.63**	5	73.13
4	5	4 943.87	**5**	**468.66**
5	5	9 001.10	**5**	**1 627.76**
6	2	31 384.31	**3**	**10 089.66**
7	1	103 191.92	**2**	**122 299.54**
8	0	∞	0	∞

Table 2: Number of instances solved and average solution times for instances on a gas network topology with 592 nodes and a varying number of intervals per arc, using the multiple-choice method

let an interval domain $D = [l_D, u_D]$ for the flow value and breakpoints $B_1 = l_D, B_2, \ldots, B_n, B_{n+1} = u_D$ be given. We use continuous $[0,1]$-variables δ_i, binary variables y_i, and the constraint

$$q = B_1 y_1 + \sum_{i=1}^{n} (B_{i+1} - B_i)\delta_i$$

together with the *filling condition* constraints $y_i \geq \delta_i, i = 1, \ldots, n$ and $\delta_i \geq y_{i+1}, i = 1, \ldots, n-1, \delta_n \geq 0$. A piecewise-linear function f of q can then be written as

$$f(q) = f(B_1)y_1 + \sum_{i=1}^{n} (f(B_{i+1}) - f(B_i))\delta_i.$$

By construction, the binary y-variables are decreasing, where a "jump" $y_i = 1$, $y_{i+1} = 0$ means that the flow q lies in the i-th interval $[B_i, B_{i+1}]$. In fact, the binary variables used by the incremental method in order to encode the active interval exactly match those from Model 9. This suggests using the incremental method as a linearization method when testing our dual-flow formulation.

The incremental method is widely used in practice and has proved very useful in the context of gas networks, see e.g. [CPSM14]. Like MCM, it leads to locally ideal formulations, but of course also only in case of a single arc. In the following, we compare the standard formulation with and without adding constraints from the totally unimodular dual-flow formulation.

The results can be found in Table 3. Using the incremental method reduces the overall runtime by a large factor, such that instances with up to 12 intervals per arc (20 with the DF-formulation on paths) can now be solved. This agrees with [CPSM14], where a recent in-depth computational study for piecewise-linear functions in the context of gas network optimization sees the incremental method come out on top, outperforming the multiple-choice method by several orders of magnitude for some test sets. Recognizing this interesting connection to the incremental method also provides an additional argument for the dual-flow formulation, as its variables seem to suit solvers well in this context. Providing the solver with the DF-formulation on paths again increases the performance of the solver significantly. For further related computational experiments see [LM16].

The results of this subsection show that the DF-formulation can have a large benefit, not only if the feasible set – as in the last subsection – can be described as (CPMCS) as

# intervals per arc	INC solved	INC CPU[s]	INC + DF-paths solved	INC + DF-paths CPU[s]
4	**5**	**5.61**	5	6.10
5	5	13.73	**5**	**10.50**
6	5	141.02	**5**	**41.96**
7	5	197.94	**5**	**68.49**
8	5	1424.02	**5**	**195.95**
9	5	1144.44	**5**	**857.59**
10	5	25506.75	**5**	**837.45**
12	3	85712.83	**5**	**3048.45**
15	0	∞	**5**	**44275.51**
20	0	∞	**1**	**824.18**
25	0	∞	0	∞

Table 3: Number of instances solved and average solution times for instances on a gas network topology with 592 nodes and a varying number of intervals per arc, using the incremental method

a whole, but also if (CPMCS) is present as a substructure.

5 Conclusion

In this paper, we have introduced the clique problem with multiple-choice constraints along with an important special case which we name staircase compatibility. It generalizes compatibility structures known from different areas of application, such as project scheduling and piecewise linearization. We showed that the convex hull of feasible solutions to this problem can be described by a totally unimodular constraint matrix of polynomial size if the compatibility graph has this particular structure. Furthermore, we showed that the constraint matrix is then cographic, which yields a dual-flow formulation for the problem.

For two example applications, where (CPMCS) is present as a substructure, we showed that using totally unimodular formulations for (CPMCS) represents a significant improvement over a naive general formulation and can vastly reduce solution time.

With these insights, future research may aim to identify (CPMCS) within more applications or even automatically in general MIPs in order to do a reformulation. Connected to this is the question whether – or in which cases – staircase compatibility structure can be recognized from a compatibility graph if the partition is not given beforehand. This might also be interesting from a graph-theoretic point of view.

Acknowledgements

We gratefully acknowledge the computing resources provided by the group of Michael Jünger in Cologne. In particular, we thank Thomas Lange for his technical support. We also thank Rodrigo Alexander Castro Campos, Sergio Luis Pérez Pérez, Gualberto

Vazquez Casas and Francisco Javier Zaragoza Martínez for our fruitful discussions on the topic. Furthermore, we acknowledge financial support by the BMBF under grant 05M13WEE. Moreover, we thank the EnCN for support within research focus Simulation, Project TP6, and the DFG for their support within Projects A05, B06 and B07 in CRC TRR 154. Last but not least, we thank the anonymous referee for his help in improving this paper.

References

[BGM18] Andreas Bärmann, Patrick Gemander, and Maximilian Merkert. The clique problem with multiple-choice constraints under a cycle-free dependency graph. `http://www.optimization-online.org/DB_HTML/2018/01/6436.html`, 2018.

[BMS17] Andreas Bärmann, Alexander Martin, and Oskar Schneider. A comparison of performance metrics for balancing the power consumption of trains in a railway network by slight timetable adaptation. *Public Transport*, 9(1-2):95–113, 2017.

[CPSM14] Carlos M. Correa-Posada and Pedro Sánchez-Martín. Gas network optimization: A comparison of piecewise linear models. `http://www.optimization-online.org/DB_FILE/2014/10/4580.pdf`, 2014.

[Fou84] Robert Fourer. Staircase matrices and systems. *SIAM Review*, 26(1):1–70, 1984.

[GH62] Alain Ghouila-Houri. Caracterisation des matrices totalement unimodulaires. *Comptes Rendus Hebdomadaires des Séances de l'Académie des Sciences (Paris)*, 254(1):1192–1194, 1962.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York, 1979.

[GMMS12] Björn Geißler, Alexander Martin, Antonio Morsi, and Lars Schewe. Using piecewise linear functions for solving MINLPs. In Jon Lee and Sven Leyffer, editors, *Mixed Integer Nonlinear Programming*, volume 154 of *The IMA Volumes in Mathematics and its Applications*, pages 287–314. Springer New York, 2012.

[Gur17] Gurobi Optimization, Inc. Gurobi optimizer reference manual. `http://www.gurobi.com`, 2017.

[Hel23] Ed. Helly. Über Mengen konvexer Körper mit gemeinschaftlichen Punkten. *Jahresbericht der Deutschen Mathematiker-Vereinigung*, 32:175–176, 1923.

[JL84] Robert G. Jeroslow and James K. Lowe. Modelling with integer variables. In Bernhard Korte and Klaus Ritter, editors, *Mathematical Programming at Oberwolfach II*, volume 22 of *Mathematical Programming Studies*, pages 167–184. Springer Berlin Heidelberg, 1984.

[KHPS15] Thorsten Koch, Benjamin Hiller, Marc Pfetsch, and Lars Schewe, editors. *Evaluating Gas Network Capacities*. MOS-SIAM Series on Optimization, 2015.

[LM16] Frauke Liers and Maximilian Merkert. Structural investigation of piecewise linearized network flow problems. *SIAM Journal on Optimization*, 26(4):2863–2886, 2016.

[Mer17] Maximilian Merkert. *Solving Mixed-Integer Linear and Nonlinear Network Optimization Problems by Local Reformulations and Relaxations*. Doctoral thesis, Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), 2017.

[MM57] Harry M. Markowitz and Alan S. Manne. On the solution of discrete programming problems. *Econometrica*, 25(1):84–110, 1957.

[MSSU01] Rolf H. Möhring, Andreas S. Schulz, Frederik Stork, and Marc Uetz. On project scheduling with irregular starting time costs. *Operations Research Letters*, 28(4):149–154, 2001.

[Oxl06] James G. Oxley. *Matroid Theory (Oxford Graduate Texts in Mathematics)*. Oxford University Press, Inc., New York, NY, USA, 2006.

[Pac16] Jörn Pachl. *Systemtechnik des Schienenverkehrs: Bahnbetrieb planen, steuern und sichern*. Springer Vieweg, 2016.

[Pad00] Manfred Padberg. Approximating separable nonlinear functions via mixed zero-one programs. *Operations Research Letters*, 27(1):1–5, August 2000.

[PFG+15] Marc E. Pfetsch, Armin Fügenschuh, Björn Geißler, Nina Geißler, Ralf Gollmer, Benjamin Hiller, Jesco Humpola, Thorsten Koch, Thomas Lehmann, Alexander Martin, Antonio Morsi, Jessica Rövekamp, Lars Schewe, Martin Schmidt, Rüdiger Schultz, Robert Schwarz, Jonas Schweiger, Claudia Stangl, Marc C. Steinbach, Stefan Vigerske, and Bernhard M. Willert. Validation of nominations in gas network optimization: Models, methods, and solutions. *Optimization Methods and Software*, 30(1):15–53, 2015.

[Sch98] Alexander Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons Ltd, 1998.

[Sey80] P.D Seymour. Decomposition of regular matroids. *Journal of Combinatorial Theory, Series B*, 28(3):305 – 359, 1980.

[SZ15] Christoph Schwindt and Jürgen Zimmermann, editors. *Handbook on Project Scheduling (Vol. 1 + Vol. 2)*. Springer, 2015.

[Vie15] Juan Pablo Vielma. Mixed integer linear programming formulation techniques. *SIAM Review*, 57(1):3–57, 2015.