Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

M. E. Pfetsch[A], A. Fügenschuh[B], B. Geißler[C],
N. Geißler[D], R. Gollmer[E], B. Hiller[B], J. Humpola[B],
T. Koch[B], T. Lehmann[B], A. Martin[C], A. Morsi[C],
J. Rövekamp[D], L. Schewe[C], M. Schmidt[F], R. Schultz[E],
R. Schwarz[B], J. Schweiger[B], C. Stangl[E],
M. C. Steinbach[F], S. Vigerske[B], B. M. Willert[F]

Validation of Nominations in Gas Network Optimization: Models, Methods, and Solutions

[A]Technische Universität Darmstadt, Germany
[B]Zuse Institute Berlin, Germany
[C]Friedrich-Alexander Universität Erlangen-Nürnberg, Germany
[D]Open Grid Europe GmbH, Essen, Germany
[E]Universität Duisburg-Essen, Germany
[F]Leibniz Universität Hannover, Germany

ZIB-Report 12-41 (November 2012)

Herausgegeben vom
Konrad-Zuse-Zentrum für Informationstechnik Berlin
Takustraße 7
D-14195 Berlin-Dahlem

Telefon: 030-84185-0
Telefax: 030-84185-125

e-mail: bibliothek@zib.de
URL: http://www.zib.de

ZIB-Report (Print) ISSN 1438-0064
ZIB-Report (Internet) ISSN 2192-7782

Validation of Nominations in Gas Network Optimization: Models, Methods, and Solutions

Marc E. Pfetsch[A] Armin Fügenschuh[B] Björn Geißler[C]
Nina Geißler[D] Ralf Gollmer[E] Benjamin Hiller[B]
Jesco Humpola[B] Thorsten Koch[B] Thomas Lehmann[B]
Alexander Martin[C] Antonio Morsi[C] Jessica Rövekamp[D]
Lars Schewe[C] Martin Schmidt[F] Rüdiger Schultz[E]
Robert Schwarz[B] Jonas Schweiger[B] Claudia Stangl[E]
Marc C. Steinbach[F] Stefan Vigerske[B]
Bernhard M. Willert[F]

[A]Technische Universität Darmstadt, Germany
[B]Zuse Institute Berlin, Germany
[C]Friedrich-Alexander Universität Erlangen-Nürnberg, Germany
[D]Open Grid Europe GmbH, Essen, Germany
[E]Universität Duisburg-Essen, Germany
[F]Leibniz Universität Hannover, Germany

Abstract

In this article we investigate methods to solve a fundamental task in gas transportation, namely the *validation of nomination problem*: Given a gas transmission network consisting of passive pipelines and active, controllable elements and given an amount of gas at every entry and exit point of the network, find operational settings for all active elements such that there exists a network state meeting all physical, technical, and legal constraints.

We describe a two-stage approach to solve the resulting complex and numerically difficult mixed-integer non-convex nonlinear feasibility problem. The first phase consists of four distinct algorithms facilitating mixed-integer linear, mixed-integer nonlinear, reduced nonlinear, and complementarity constrained methods to compute possible settings for the discrete decisions. The second phase employs a precise continuous nonlinear programming model of the gas network. Using this setup, we are able to compute high quality solutions to real-world industrial instances whose size is significantly larger than networks that have appeared in the literature previously.

1 Introduction

Natural gas is an energy resource of paramount importance for human society. In Europe it accounts for about 25% of the primary energy consumption and is distributed through a pipeline network with a total length of more than 100,000 km. The reliable and efficient operation of these pipeline networks is a permanent challenge asking for computer-based automated decision support.

The European legislative framework for gas transportation has undergone significant changes in the recent past. By European-Union regulation [25], gas trading and transportation now have to be unbundled, i.e., performed by independent companies. Formerly, an integrated gas company, possibly owning the pipelines, storages, and consuming power stations could conclude long-term supplier contracts and then operate its network in a dedicated point-to-point fashion. Today, gas trading takes place at virtual, liberalized, and non-discriminatory markets, with consumers, suppliers, storage, and transportation handled by different independent players. The network itself is owned and operated by so-called *gas transmission system operators* (TSO), who on behalf of gas trading companies have to ensure the delivery of the traded gas from suppliers to consumers.

The market participants can obtain rights (contracts) from a TSO to supply or consume gas from the network at given entries or exits. The TSO decides on the total capacity of the rights sold for any particular entry or exit. Note that the rights for entries and exits are sold independently from each other.

For any group of entries and exits the owners of these rights can *nominate* any balanced[1] amount of gas up to the contractual limit to be transported. When selling the rights to nominate at single points, the TSO already warranted that they are capable of fulfilling any such combined transportation request. The amount of nominated gas of all entry and exit points of the network constitutes a *nomination*, i.e., the total amount of gas that has to be transported at a given time. It further includes requirements on the gas composition and pressure bounds. For more details of the German legislation, see [10].

In the present article we investigate a fundamental problem in gas transportation, namely the *validation of nomination problem* (NoVa): Given a gas transmission network consisting of passive pipelines and active, controllable elements such as valves and compressors, and given a nomination (an amount of gas) at every entry and exit point, the task is to find an operational setting of the active elements of this network such that there exists a network state meeting all physical, technical, and legal constraints.

In practice, gas networks are operated/dispatched in a transient manner, i.e., over a continuous time horizon. A transient model of a gas network, thus, would require as input both initial states and future nomination profiles in continuous time. Since we consider mid to long term planning, both are unknown. Thus, in the present paper we exclusively deal with stationary (steady state) gas transportation.

Of course, a gas transportation operator must avoid granting rights of nominations that are legally correct but technically infeasible. This is closely related to NoVa. Since it is clearly desirable to grant as many nomination rights as possible, it is important to detect nominations that cannot be handled by the network.

Altogether, NoVa leads to a complex mixed-integer non-convex nonlinear feasibility problem, for which new solution approaches are proposed in the present paper.

[1] In fact, unbalanced nominations are allowed, but they have to be turned into balanced ones by using regulating energy, which incurs additional cost.

The state-of-the-art in practice is the verification of a given nomination by using simulation tools, i.e., an experienced planner tries to find suitable settings of the active elements by hand and uses a gas network simulator to test the outcome. With the models described in the present paper we aim at automatizing this procedure. Furthermore, if the planner fails to find a suitable setting, it does not prove its non-existence. This is different for some of the models presented here, which can be solved to global optimality, thus proving infeasibility in case the solver is not able to find any feasible solution.

1.1 Outline of the Paper

Since NoVa involves discrete decisions as well as nonlinearities, it is typically very hard to solve larger (real-world) instances. We propose a sequential approach. In Section 3, we present four methods that aim at finding good discrete decisions, while approximating the nonlinearities. Then, we use these decisions (and corresponding network states) as input for a nonlinear optimization model that includes a detailed physical model, see Section 3.5. In this way, we can obtain high quality solutions.

In Sections 4 and 5, we present an extensive computational study. In Section 4, we first evaluate each of the four approaches via computations on real-world instances and discuss their different features. It turns out that each approach has its strengths in different areas. In Section 5, we discuss the combination of these four approaches yielding a fairly successful solver for the NoVa problem. To the best of our knowledge, the successful solution of gas transportation problems of this size and complexity has never been reported in the literature, so far.

1.2 Related Literature and Our Contributions

In this section, we will briefly highlight some of the related literature. For a recent survey on gas network optimization in general, we refer to [64].

Variants of NoVa, mostly for minimizing operating costs, have been studied by many researchers. Early approaches were often based on dynamic programming (DP). In [82], DP was applied to steady state gas network optimization in cases where the underlying gas network consists of a single straight line; later, branched network structures were considered in [86]. A similar approach was described in [47], before [37, 9] studied networks containing branches and loops of arbitrary size. A detailed overview on DP approaches to gas network optimization can be found in [16].

Since the NoVa problem contains nonlinearities, nonlinear programming (NLP) techniques were applied, too. In [43, 46, 79], subgradient methods were used to tackle the problem. Sequential linear programming techniques were used in [21]. Sequential quadratic programming techniques were applied in [29, 24]. The problem was also studied with respect to primal-dual interior point methods [73]. In [74], locally linearized mixed-integer nonlinear programs (MINLPs) in combination with a receding horizon technique are used. Interval analysis techniques were used in [8] to minimize operation costs for instances without discrete decisions on the basis of the Belgian network.

Several papers deal with the extension or dimensioning of existing gas transportation networks using NLP/MINLP techniques. They implicitly handle NoVa problems via (simplified) gas transportation models, see, e.g., [2, 3, 20, 40, 85].

Mixed-integer linear programming (MILP) methods have already been successfully applied to steady state optimization of gas networks in [56, 55] and more recently also to the transient case [57, 52, 30, 35, 34]. A combination of both, MILP- and NLP-techniques, was suggested in [22].

Other techniques to tackle transient gas network optimization problems include simulated annealing [83, 51], genetic algorithms [48], ant colony optimization [18], and hierarchical system theory [59]. The operating cost minimization problem can also be formulated as a non-cooperative game, where compressors and entries are the players, and communication is established through the network connectivity constraints. The solution is then given as a Nash-equilibrium found by an iterative algorithm, see, e.g., [62, 63].

The idea of variable elimination in models for flows adhering to Kirchhoff's first and second laws, which we elaborate in Section 3.3, can be traced back at least to [39] and was picked up repeatedly later on, see [53], [65], and the monograph [60]. In our approach, this variable elimination is just a building block in a concerted feasibility testing in large meshed gas distribution networks with compressors, resistors, and control valves, i.e., elements whose control involves combinatorial decisions.

Unlike the existing literature, our approach focuses on discrete decisions and provides a higher level of (practical) detail. The MILP approach, which we present in Section 3.1, builds on our own developments and extends it towards solutions that can successfully be validated with a sophisticated gas network simulation. The spatial branching approach of Section 3.2 applies outer approximations techniques and adapts them to gas networks; the approach is based on a new approximation of compressors (compressor groups). The RedNLP approach of Section 3.3 provides a novel transshipment heuristic to find good discrete decisions. The MPEC approach is, as far as we know, the first application of complementarity constraints in this area and applies a new two-stage solution approach. Finally, the combination of these four approaches with a validation by a detailed NLP provides to the best of our knowledge the best solution technique to date.

2 Detailed Physical Model

In this section, we describe a detailed physical and technical model of the gas transportation network and its components.

We model a gas network as a directed graph $G = (V, A)$ with nodes representing junctions of network elements. The arcs represent pipes, compressor groups, valves, control valves, and resistors, denoted by A_{pipe}, A_{cg}, A_{va}, A_{cv}, A_{rs}, respectively. The set of arcs can be divided into active (A_{cg}, A_{va}, A_{cv}) and passive (A_{pipe},A_{rs}) ones. Active elements can be controlled directly and have several states of operation. This is not the case for passive arcs. We will treat each type in the presentation below.

For each node u, we introduce a gas pressure variable $p_u \geq 0$. Moreover, $q_a \in \mathbb{R}$ denotes the variable of the mass flow along arc a and describes the mass of gas passing through a given area per unit of time. We use the convention that $q_a > 0$ refers to gas flow in the direction of the arc and $q_a < 0$ indicates that the gas flows in the opposite direction. For each network element $a = (u, v)$, there is a relation between the mass flow q_a and the pressures p_u and p_v, depending on the type of a; see below for details. Due to technical limitations or contractual requirements, we may have pressure bounds

$\underline{p}_u \le p_u \le \overline{p}_u$ for each node u and mass flow bounds $\underline{q}_a \le q_a \le \overline{q}_a$ for each arc a; note that $\underline{q}_a$ and $\overline{q}_a$ are allowed to be negative due to the direction of the flow.

We need the following two additional flow quantities: The volumetric flow $Q = q/\rho$ describes the volume of gas passing through a given area per time and depends on the gas density ρ. The normal volumetric flow $Q_0 = q/\rho_0$ is the volumetric flow under normal density ρ_0, based on normal pressure $p_0 = 101\,325\,\mathrm{Pa}$ and normal temperature $T_0 = 273.15\,\mathrm{K}$.

At some nodes gas is supplied into the network, while it is discharged at other nodes. The mass flow $\boldsymbol{d} = (d_u)_{u \in V}$ arising from a nomination specifies the amount of gas entering ($d_u \ge 0$) or leaving ($d_u \le 0$) the network at each node u. At each junction, the gas flow is subject to the mass flow conservation condition

$$\sum_{a \in \delta^+(u)} q_a - \sum_{a \in \delta^-(u)} q_a = d_u \qquad \forall\, u \in V, \tag{1}$$

where $\delta^+(u)$ and $\delta^-(u)$ are the arcs leaving and entering node u, respectively. In this paper, we assume a *homogeneous gas composition*, i.e., we approximate the molar mass m, pseudocritical pressure p_c, pseudocritical temperature T_c, normal density ρ_0, and compressibility factor under normal conditions $z_0 = z(p_0, T_0)$ by constants. In this case, (1) can equivalently be expressed in Q_0 instead of q.

2.1 Modeling of Pipes

Most network elements are pipes, and they are the only elements with a significant length. A pipe $a \in A_{\text{pipe}}$ is a passive network element, i.e., the gas flow cannot be controlled directly. Instead, it results from the laws of physics, which are described by a system of partial differential equations: the continuity, momentum, and energy equation, e.g., [50, 26]. In the stationary case these equations can be simplified to a set of ordinary differential equations:

$$\frac{\partial q_a}{\partial x} = 0, \tag{2}$$

$$\frac{\partial p_a}{\partial x} + \frac{q_a^2}{A_a^2}\frac{\partial}{\partial x}\frac{1}{\rho} + g\,\rho\,\frac{(h_v - h_u)}{L_a} + \lambda_a(q_a)\frac{|q_a|\,q_a}{2A_a^2\,D_a\,\rho} = 0, \tag{3}$$

$$q_a\,c_p\,\frac{\partial T}{\partial x} - \frac{q_a T}{\rho\, z}\frac{\partial z}{\partial T}\frac{\partial p}{\partial x} + g\,q_a\,\frac{(h_v - h_u)}{L_a} + \pi\,D_a\,c_{\text{HT}}\,(T - T_{\text{soil}}) = 0. \tag{4}$$

Here, x denotes the one-dimensional position along the pipe and T the gas temperature. The parameters L_a, D_a, and A_a specify the length, diameter, and cross-sectional area of the pipe a. Further, h_u and h_v are the normal heights of the nodes u and v, and g is the gravitational acceleration constant. The parameters c_{HT}, c_p, and T_{soil} denote the heat transfer coefficient, the specific heat capacity, and the soil temperature, respectively. The compressibility factor $z = z(p, T)$ is an approximation for the deviation of real gas from ideal gas. There exists no exact model, but several approximations. We use the formula of the American Gas Association (AGA), see, e.g., [49], in this paper:

$$z(p, T) = 1 + 0.257\,\frac{p}{p_c} - 0.533\,\frac{p/p_c}{T/T_c}. \tag{5}$$

See Papay [61] for another approximation. Finally, $\lambda_a(q_a)$ is the friction factor that strongly depends on the vorticity of the flow. For *laminar* flow (i.e., small mass flow rates), the friction factor complies with the exact model of Hagen-Poiseuille, see, e.g., [27]:

$$\lambda_a^{\mathrm{HP}}(q_a) = \frac{64}{Re(q_a)}. \tag{6}$$

With $Re(q_a)$ we denote the Reynolds number, which is a measure of the ratio of inertial forces to viscous forces and is also used to characterize the different flow regimes, i.e., laminar or turbulent flow. The Reynolds number $Re(q_a)$ is linear in $|q_a|$. For *turbulent* flow (i.e., large mass flow rates) no exact model is known. A highly accurate empirical model is the one of Prandtl-Colebrook, see, e.g., [19] and [66, Chap. 9]:

$$\frac{1}{\sqrt{\lambda_a^{\mathrm{PC}}(q_a)}} = -2\log_{10}\left(\frac{2.51}{Re(q_a)\sqrt{\lambda_a^{\mathrm{PC}}(q_a)}} + \frac{k_a}{3.71\,D_a}\right), \tag{7}$$

where k_a denotes the (integral) roughness of the pipe.

In addition to the differential equations, everywhere in the network pressure, temperature, and density are coupled, see, e.g., [50, 58]. Several models exist for this equation of state. A common choice is the thermodynamical standard equation:

$$\rho = \frac{m}{R}\frac{p}{Tz(p,T)}. \tag{8}$$

In our stationary model, the continuity equation (2) reduces to the fact that the gas flow along a pipe is constant, which we already ensured by introducing a single mass flow variable for each pipe. Moreover, we assume a *constant gas temperature* T throughout this paper. Hence, the energy equation (4) can be neglected. This leaves us with the momentum equation (3).

2.2 Modeling of (Control) Valves

Valves are active elements used to control the gas flow. A closed valve prevents gas from flowing through this arc, decoupling the pressures at both sides of the valve. If the valve is open, however, no restriction on the flow through the valve applies. Since the physical length of a valve is negligible, we assume the pressures at both ends to be equal in this case. For a valve $a = (u, v) \in A_{\mathrm{va}}$, we introduce a binary variable s_a to obtain

$$\begin{aligned} s_a = 0 \quad &\Longrightarrow \quad q_a = 0, &&p_u,\, p_v \text{ arbitrary}, \\ s_a = 1 \quad &\Longrightarrow \quad q_a \text{ arbitrary}, &&p_u = p_v. \end{aligned} \tag{9}$$

Control valves extend the model of valves by allowing to reduce the pressure within certain technical limits. They are usually found at the interconnection points of subnetworks with different pressure ranges. For a control valve $a = (u, v) \in A_{\mathrm{cv}}$, we assume that the pressure reduction $\Delta_a = p_u - p_v \geq 0$ lies within $[\underline{\Delta}_a, \overline{\Delta}_a]$. The model of a valve is extended by an additional binary variable s_a^{act}, which determines whether the control valve is actively working or not. We obtain the following model:

$$s_a^{\mathrm{act}} = 1 \quad \Longrightarrow \quad 0 \leq \underline{\Delta}_a \leq \Delta_a \leq \overline{\Delta}_a, \quad q_a \geq 0, \quad s_a = 1. \tag{10}$$

This allows to model the following three control valve states: "closed" ($s_a = 0$, $s_a^{\mathrm{act}} = 0$), "active" ($s_a = 1$, $s_a^{\mathrm{act}} = 1$), and "bypass" ($s_a = 1$, $s_a^{\mathrm{act}} = 0$).

2.3 Modeling of Compressor Groups

While the gas pressure decreases along a pipe, groups of compressors allow to increase the pressure to facilitate long-distance gas transmission. Similar to control valves, a compressor group $a = (u, v) \in A_{\mathrm{cg}}$ can take three states "closed", "active", and "bypass". Thus, we add a binary variable s_a together with (9) and a binary variable s_a^{act} that controls whether a compressor group is active, i.e., compressing.

Each compressor group consists of at least one compressor, which increases the pressure of the gas, and additional technical devices. The set of *configurations* of a compressor group defines the valid combinations of the included compressors. For instance, if high throughput and moderate pressure increase is required, the compressor units may work in parallel, while in situations with moderate pressure increase some compressors may be deactivated. The configurations introduce an additional discrete aspect to the model of compressor groups. It is straightforward to model this by using further discrete variables.

The modeling of compressors is addressed, for instance, in [15, 14, 84]. The feasible range of the pressure increase of a compressor depends on technical parameters and the volumetric flow. For a compressor, we call the set of all tuples (p_u, p_v, Q_a), such that the compressor can increase the pressure from p_u to p_v for volumetric flow Q_a, its *operation range*. This is a nonlinearly bounded non-convex set, which is described using the so-called characteristic diagram of the compressor; see Figure 3 for an example.

The energy needed to compress a unit of gas from p_u to p_v is described by the adiabatic head

$$H_{\mathrm{ad},a}(p_u, p_v) = z(p_u, T)\, T\, \frac{R}{m} \frac{\kappa}{\kappa - 1} \left(\left(\frac{p_v}{p_u} \right)^{(\kappa-1)/\kappa} - 1 \right), \tag{11}$$

with the universal gas constant R and isentropic exponent κ, see, e.g., [17]. The power consumption of an active compressor is

$$P_a = \frac{H_{\mathrm{ad},a}}{\eta_{\mathrm{ad},a}}\, q_a, \tag{12}$$

where $\eta_{\mathrm{ad},a}$ denotes the adiabatic efficiency of a compressor.

Often, larger subnetworks of compressor groups, control valves, and valves are located at the intersection of several pipelines. The topology of such a subnetwork can be used to realize many operation modes by appropriately setting the active elements of this subnetwork. In these cases, usually very few of the possible switching combinations correspond to realistic operation modes. Thus, for such subnetworks our model includes additional linear constraints with discrete variables, enforcing so-called *Subnetwork Operation Modes (SOM)*, ensuring that only one of the permitted switching combinations is used.

2.4 Modeling of Additional Pressure Losses

Besides the pressure loss resulting from friction of the flow through the pipes, there are some properties and components of the network inducing an additional pressure loss. Such a pressure loss is caused by, e.g., flow diversion and turbulences in shaped pieces, measurement devices, curved parts within compressor groups, filter systems, measurement devices, reduced radii, and partially closed valves. Neither exact data nor exact models are available for most of these pressure loss effects. We represent them by inserting a virtual *resistor* element at affected locations of the network. Two different types of resistor models are used: either a constant pressure loss in flow direction

$$p_u - p_v = \Delta_a \operatorname{sgn}(q_a), \tag{13}$$

or a pressure loss according to an equation of Darcy-Weisbach type using the parameters drag factor ζ_a and a fictitious diameter D_a:

$$p_u - p_v = \frac{8\zeta_a}{\pi^2 D_a^4} \frac{|q_a|\, q_a}{\rho_u}. \tag{14}$$

At this point, we have proposed models for each network component, enabling us to formulate the nomination validation problem NoVa as a feasibility problem: Given a gas network $G = (V, A)$ as defined above, can one realize a given nomination $\boldsymbol{d}$?

2.5 Model Adjustments and Approximations

A model consisting of the presented component models leads to a discrete-continuous, nonlinear, nonsmooth feasibility problem with ordinary differential equations. Due to its complexity, this model cannot be solved directly for large-scale instances. Thus, some aspects of this detailed model are approximated in order to yield more accessible models. We present several such approximations that are used by the approaches presented in Section 3.

The momentum equation (3) is replaced by the following approximation from the literature as an approximation of the behavior of each pipe $a = (u, v) \in A_{\text{pipe}}$, see, e.g., [5, 50]:

$$p_v^2 = \left(p_u^2 - \Lambda_a(q_a)\,|q_a|\,q_a \frac{e^{S_a} - 1}{S_a}\right) e^{-S_a}, \tag{15}$$

with

$$S_a = \frac{2g\,(h_v - h_u)\,m}{R\,z_m\,T}, \tag{16}$$

$$\Lambda_a(q_a) = \left(\frac{4}{\pi}\right)^2 \frac{L_a}{D_a^5} \frac{m}{R} z_m\,T\,\lambda_a(q_a), \tag{17}$$

$$z_m = z(p_m, T), \tag{18}$$

$$p_m = \frac{2}{3}\left(p_u + p_v - \frac{p_u\,p_v}{p_u + p_v}\right). \tag{19}$$

The influence of the slope of a pipe is summarized in S_a according to (16). Under certain assumptions, this formula gives an analytic solution to the momentum equation.

It turns out that even the simplified models as just described cannot be solved for real-world instances with state-of-the-art black-box solvers, as we will demonstrate in Section 4.1. In the following section, we therefore present a new two-step approach for NoVa.

3 Our Solution Approach

In this section, we present an approach to solve NoVa that works in two steps: In the first step, we apply four approaches to handle the discrete decisions necessary to solve NoVa. These models use an approximation for each network component. The solutions provided by each of these approaches are validated in a second step to find accurate solutions for NoVa by fixing the discrete decisions and using the solutions as a starting point for solving in a more accurate reference NLP model.

Of course, both the choice of a model and the choice of an approach imply certain tradeoffs regarding physical accuracy and computational tractability. Nevertheless, it is clear that a model that is sufficiently accurate for practical application will feature both discrete decisions and non-convex relationships between pressure and flow, i.e., one has to solve a MINLP problem.

3.1 The MILP Approach

Whereas state-of-the-art solvers for mixed-integer linear programming problems are often able to solve even problems with millions of variables and constraints in relatively short time, algorithms for solving non-convex MINLPs are in most cases not yet able to treat large-scale problems as they appear in gas networks (see Section 4.1 below). Thus, we exploit the strength of MILP solvers for NoVa via linearization of the nonlinearities.

We start with the model parts and formulas given in Section 2. Here, we have the basic variables $\boldsymbol{p}$, $\boldsymbol{Q_0}$, $\boldsymbol{P}$ and $\boldsymbol{s}$. Compared to constraints involving nonlinearities, combinatorial aspects arising from valves, control valves, compressors, or from subnetwork operation modes (SOM) are fairly straightforward to integrate within a MILP model. In the following, we thus focus on the handling of the nonlinearities arising from pipes and compressors.

To model a pipe $a = (u, v) \in A_{\text{pipe}}$, we assume a constant (mean) compressibility factor z and a constant friction factor λ_a. In particular, we use

$$z_m = z\Big(\frac{\min\{\underline{p}_u, \underline{p}_v\} + \max\{\overline{p}_u, \overline{p}_v\}}{2}, T\Big), \tag{20}$$

$$\lambda_a = \Big(2\log_{10}\Big(\frac{D_a}{k_a}\Big) + 1.138\Big)^{-2}. \tag{21}$$

instead of (18)–(19) and (6)–(7). Under the assumption of a constant compressibility factor, Equation (15) is separable and can hence be expressed by the sum of three nonlinear univariate functions. The power consumption of a compressor (see (11) and (12)) is modeled by a recursive decomposition into three nonlinear univariate functions and one bivariate product. We remark here that from a theoretical point of view this decomposition is not necessary for our approach, but it is very valuable for the application to large-scale instances.

Our approach to integrate the nonlinearities in a MILP model is based on piecewise polyhedral relaxations of the nonlinearities, i.e., the nonconvexities arising from nonlinear equality constraints in the MINLP model are transformed into nonconvexities expressed by linear constraints and integrality conditions using a set of additional variables.

Since all basic variables (mainly $\boldsymbol{p}$ and $\boldsymbol{Q_0}$) are bounded, we can assume that any occurring nonlinear function $f : D \to \mathbb{R}$ is defined on a compact set $D \subset \mathbb{R}^d$, typically a box. We triangulate D into simplices $S_1, \ldots, S_n$ and approximate f by a piecewise linear interpolation $g_i(x)$ for $x \in S_i$, g_i affine, $i = 1, \ldots, n$, over the vertices of the simplices $S_1, \ldots, S_n$. Additionally, we compute upper bounds ϵ_i, $i = 1, \ldots, n$, for the approximation error on each simplex and replace each occurrence of f in the constraint set by its piecewise polyhedral outer approximation. The interpolations together with the bounds for the approximation errors are computed with an adaptive refinement algorithm, which makes use of convex underestimators in order to provide reliable error bounds. For a detailed description of the procedure we refer to [34, 33].

Finally, such a piecewise polyhedral outer approximation is modeled in terms of mixed-integer linear constraints by means of a modified version of the so-called incremental method of [54]. The necessary modifications of the method have also been introduced in [34, 33]. A crucial assumption for the application of the incremental approach is to order the simplices according to a Hamiltonian path in the dual graph of the triangulation. Simultaneously, an appropriate ordering of the vertices of each simplex must be constructed such that the first vertex of a simplex is equal to the last vertex of the predecessor simplex in the Hamiltonian path. While this is trivial in dimension one, we refer to [34, 33] for more details on how to obtain such an ordering in higher dimension.

The modified version of the incremental model is as follows:

$$x = \hat{x}_0^1 + \sum_{i=1}^{n} \sum_{j=1}^{d} \left(\hat{x}_j^i - \hat{x}_0^i\right) \delta_j^i, \tag{22}$$

$$y - e = \hat{y}_0^1 + \sum_{i=1}^{n} \sum_{j=1}^{d} \left(\hat{y}_j^i - \hat{y}_0^i\right) \delta_j^i, \tag{23}$$

$$-\epsilon_1 - \sum_{i=1}^{n-1} \omega_i(\epsilon_{i+1} - \epsilon_i) \le e \le \epsilon_1 + \sum_{i=1}^{n-1} \omega_i(\epsilon_{i+1} - \epsilon_i), \tag{24}$$

$$\sum_{j=1}^{d} \delta_j^i \le 1, \qquad 1 \le i \le n, \tag{25}$$

$$\delta_j^i \ge 0, \qquad 1 \le i \le n, \qquad 1 \le j \le d, \tag{26}$$

$$\sum_{j=1}^{d} \delta_j^{i+1} \le \omega_i \le \delta_d^i \qquad 1 \le i \le n-1, \tag{27}$$

$$\omega_i \in \{0, 1\} \qquad 1 \le i \le n-1. \tag{28}$$

Here, y denotes an approximation to $f(x)$ within the piecewise polyhedral outer approximation of the graph of f. The j-th vertex of the i-th simplex S_i of the triangulation is denoted by $\hat{x}_j^i$ and $\hat{y}_j^i := f(\hat{x}_j^i)$. Constraints (24) ensure the error tolerance e over each simplex. The binary variables ω_i, $i = 1, \ldots n-1$, are used within Constraints (27)–(28) to express that if for the i-th simplex some variable $\delta_j^i > 0$, $j \in \{1, \ldots, d\}$, then $\delta_d^k = 1$ for all previous simplices $k = 1, \ldots, i-1$ in the Hamiltonian path. The model (22)–(28) is applied to the formulas (15), (11), and (12)—with simplifications as outlined in the beginning of this section to obtain 1-D and 2-D approximations, respectively. As an example, a piecewise polyhedral relaxation of the function $f(Q_{0,a}) = \alpha_a |Q_{0,a}| Q_{0,a}$ with $D = [-4, 4]$, one of the univariate nonlinearities arising from the simplification of (15), is given in Figure 1. Here the simplices are the intervals specified by the breakpoints. The

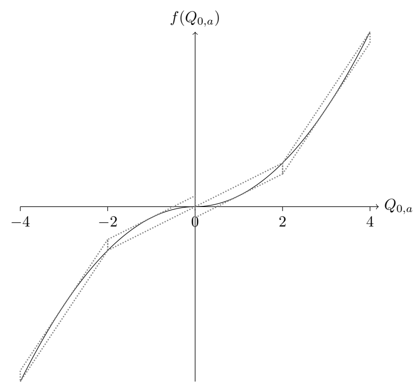


Figure 1: MILP-relaxation of $f(Q_{0,a}) = \alpha_a |Q_{0,a}| Q_{0,a}$ with $\alpha_a = 0.25$ on $[-4, 4]$ with breakpoints $-4, -2, 0, 2, 4$.

set of feasible solutions to (22)–(28) is depicted as the union of the parallelograms drawn with dotted lines in Figure 1.

Besides the function $f(Q_{0,a}) = \alpha_a |Q_{0,a}| Q_{0,a}$, the remaining univariate subexpressions arising from Equation (15) are p_u^2 and p_v^2. We introduce additional variables for these squared pressures. In many cases, however, we can then remove the original pressure variables as they are not needed, e.g., in the trivial case when the only elements incident to a node are pipes. We decide a priori whether it is necessary to introduce an approximation of p_u^2 for a node u in an optimal way using an auxiliary integer linear program (ILP). The variables of the ILP express whether a pressure variable, a squared pressure variable, or both variables are needed. The constraints then enforce these requirements. For example, for both endpoints of a pipe, squared pressure variables are needed, and similarly pressure variables are required at both endpoints of a compressor group. For a MILP formulation of a valve, however, homogeneous variables at both endpoints are sufficient. The objective of the auxiliary ILP is then to minimize the sum of variables that express that both types are required and hence to minimize the number of nonlinearities for the coupling of pressure to squared pressure variables.

Characteristic diagrams of compressors are not modeled explicitly. We enforce any active compressor to operate at least within a convex relaxation of its operation range. This is achieved by dynamically cutting off solutions via the separation of tangential hyperplanes to this convex relaxation of the characteristic diagrams, see [44]. Moreover, in order to improve the chance that the corresponding MILP solution yields a feasible solution for more accurate NLP models (see Section 3.5), we try to obtain a feasible solution by incorporating the minimization of the max norm distance to the centroids of the characteristic diagrams of all active compressors in our objective function. We remark that although operating ranges of compressors are typically not convex, the centroids of all the encountered characteristic diagrams are clearly feasible points. To choose among the different configurations of each compressor group we incorporate SOS-1 constraints on the respective binary decision variables.

3.2 The Spatial Branching (SB) Approach

The second approach solves an approximated version of NoVa with an outer approximation method. We use a tailored version of the branch-and-cut constraint integer programming framework SCIP, see [1, 71, 78], which implements a combination of convex (linear) outer approximation and spatial branching (SB) to prove global optimality. This method is also used by state-of-the-art solvers for generic MINLP problems, see, e.g., [6, 75, 76, 77, 72].

Let us give a brief review about the concept of outer approximation and spatial branching. Let $S \subseteq \mathbb{R}^n$ be the (non-convex) feasible set of the problem. A linear approximation of the feasible set is computed such that

$$S \subseteq \{x \mid Dx \leq d\}$$

for suitable $D \in \mathbb{Q}^{m \times n}$ and $d \in \mathbb{Q}^m$. This relaxation is used during the branch-and-bound algorithm and is successively refined by additional cutting planes. Branching on integer variables deals with the integrality requirements. When all integer variables take integral values, the relaxation is strengthened by recursive *spatial branching*, i.e., branching on continuous variables appearing in nonlinear terms. Spatial branching on variable i of the solution x^* to the linear relaxation refers to subdividing the previous linear relaxation into two parts

$$S \subseteq \{x \mid Dx \leq d,\ x_i \leq x_i^*\} \ \cup\ \{x \mid Dx \leq d,\ x_i \geq x_i^*\}\,.$$

For each part of the relaxation, a subproblem is created, and a tighter relaxation can be computed due to tighter variable bounds, see Figure 2 for an example. Spatial branching thus improves the convex relaxation, in particular, at places where the describing functions are non-convex. Branching is pursued until all integral variables take integral values and the convex relaxation is "close enough" to the feasible region. This way, global bounds on the objective function can be computed and the problem can be solved to global optimality.

To apply this approach to NoVa, let us first consider the modeling of pipes. We use (15) and (16) formulated in norm volumetric flow Q_0 and apply the modifications (20), (21). We introduce squared pressure variables $\pi_u := p_u^2$ for all nodes $u \in V$. Recall from Section 2 that all pressure values are nonnegative.[2] Formulating (15) using the pressure square variables yields

$$\pi_u - \beta_a \pi_v = \alpha_a\ |Q_{0,a}|\ Q_{0,a}, \tag{29}$$

for suitable constants α_a and $\beta_a > 0$, which depend on the characteristics of the pipe, see (16), (20), and (21). To separate the nonlinear part, we introduce an additional variable y_a and get

$$\pi_u - \beta_a \pi_v = y_a \tag{30}$$

$$y_a = \alpha_a\ |Q_{0,a}|\ Q_{0,a}. \tag{31}$$

While (30) is linear, (31) is a nonlinear equation and thus a non-convex constraint. The nonlinear function is handled by linear outer approximation through a special SCIP plugin

[2] The pressure in the system is never smaller than the atmospheric pressure.

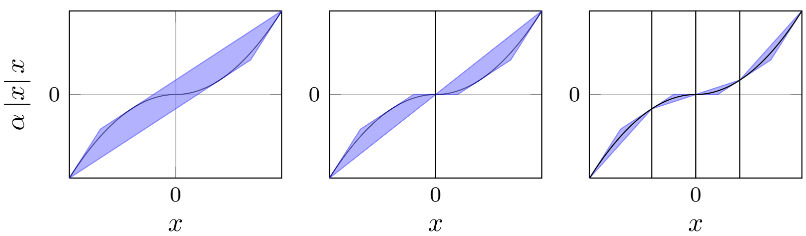


Figure 2: Successive improvement of the approximation of $x \mapsto \alpha\,|x|\,x$ by spatial branching

for this type of function performing separation of the convex hull and spatial branching, see [31, 78]. Figure 2 visualizes the treatment of the nonlinear function $x \mapsto \alpha\,|x|\,x$ by this approach.

Besides the nonlinear pressure loss equations for the pipelines, compressors are another source of nonlinear behavior. As mentioned in Section 2.3, the capability of a compressor is described by its characteristic diagram, which is represented by the set of feasible combinations of volumetric flow Q_a and adiabatic head $H_{\mathrm{ad},a}$, see Figure 3 (left). These points are obtained by physical measurements on the compressor with different pressures and flow rates. However, these quantities can not easily be computed in our model. But for a compressor $a = (u, v)$, they can approximately be mapped to the space $(p_u, p_v, Q_{0,a})$ of inlet and outlet pressure and normal flow through the compressor, respectively, via

$$\begin{pmatrix} p_u \\ p_v \\ Q_{0,a} \end{pmatrix} = p_u \begin{pmatrix} 1 \\ \left(\frac{H_{\mathrm{ad},a}}{c_2} + 1\right)^{\frac{\kappa}{\kappa-1}} \\ \frac{Q_a}{c_1} \end{pmatrix}, \tag{32}$$

where c_1 and c_2 are constants, see (5), (8), (11), and (20). Each point specifying the characteristic diagram can thus be translated into a ray of feasible combinations $\{(p_u, p_v, Q_{0,a}) \,|\, p_u \geq 0\}$ for the compressor. The convex hull of all these rays, which is a cone, approximates the operation range of the compressor. We further intersect the cone with

$$\{(p_u, p_v, Q_{0,a}) \,|\, \underline{p}_u \leq p_u \leq p_v \leq \overline{p}_v,\; \underline{Q}_{0,a} \leq Q_{0,a} \leq \overline{Q}_{0,a}\},$$

corresponding to the technical limitations of the compressor. This gives a bounded convex polyhedron. Figure 3 visualizes the different steps in this approximation procedure.

The behavior of compressor groups and their configurations is approximated in this approach. In each configuration in which several machines are involved, flow conservation constraints connect the different machines. This results in a higher dimensional polytope for the subnetwork describing the configuration (e.g., several serial or parallel compressors). Instead of using the full polytope, we project it on the boundary variables, i.e., inlet and outlet pressure and flow, by Fourier-Motzkin elimination. The result is a description of the capability of each configuration. Finally, we approximate the operation range of the group in active state by the convex hull over the polytopes of the different configurations. Closed and bypass states are modeled using additional binary variables. In our implementation, polyhedral calculations, i.e., convex hull computations and Fourier-Motzkin

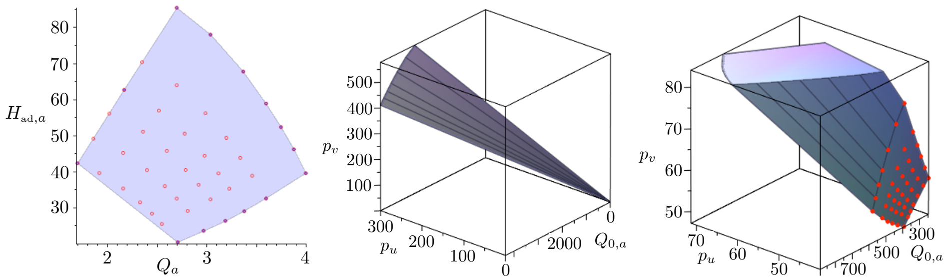


Figure 3: From the characteristic diagram of a compressor (left) over the cone (middle) to the operation range polytope (right)

elimination, are carried out by the Parma Polyhedra Library [4]. Note that the operation range polytope uses pressure variables, not squared pressure variables. Hence, explicit coupling constraints $\pi_u = p_u^2$ and $\pi_v = p_v^2$ are needed upstream and downstream of each compressor group. Algorithmically, we treat these constraints in the same way as (31).

Valves, control valves, and resistors are modeled as described in Section 2. Moreover, our approach does not involve an objective function. We therefore stop the solution process as soon as a feasible solution is found and proceed with the detailed validation, see Section 3.5.

3.3 The reduced NLP (RedNLP) Approach

The approach presented in this section relies on transforming nonlinearities into a more accessible form, reducing the problem dimension of the underlying NLP. This transformation approach is embedded into a heuristic procedure for finding promising switching decisions. The system of (linear) flow conservation (1) and (nonlinear) pipe equations (15) is transformed into an equivalent nonlinear system, where most flow and pressure variables get eliminated, because they are explicit functions of a relatively small group of variables, consisting of one flow variable per fundamental network cycle and two pressure variables per active arc. Apart from the explicit formulae for flow and pressure variables, the transformed system contains implicit equations whose number equals the number of fundamental cycles of the network and whose unknowns are just the variables from the mentioned group. The approach aims at checking feasibility for a set of switching states of active elements, either predefined or resulting from a transshipment heuristic.

Transformation of Nonlinearities For a given setting of all the active elements (*switching state*) of the network, the directed graph $\bar{G} = (V, \bar{A}) \subset G$ models the relevant network, where $\bar{A}$ denotes the set of all arcs not being in closed state.

Let $\mathcal{A}^+$ denote the node-arc-incidence-matrix and $\mathcal{A}$ be the submatrix of full row rank arising from the deletion of one row, corresponding to a preselected, pipe-incident root node $\overline{u} \in V$ with pressure variable $p_{\overline{u}}$.

Let $\boldsymbol{\pi}$ and $|\boldsymbol{Q_0}|\,\boldsymbol{Q_0}$ denote the vectors with components $\pi_u, u \in V \setminus \overline{u}$, and $|Q_{0,a}|\,Q_{0,a}$, $a \in \bar{A}$. For $a = (u, v) \in A_{\mathrm{cg}} \cup A_{\mathrm{cv}}$, i.e., for compressor groups and control valves, π_u and π_v denote squared outgoing and ingoing pressures and $\Delta_a := \pi_v - \pi_u$. For $a \notin A_{\mathrm{cg}} \cup A_{\mathrm{cv}}$, we define $\Delta_a = 0$. The diagonal matrix $\alpha = \mathrm{diag}(\Lambda_a)$ has entries which either are the pipe specific values Λ_a, $a \in A_{\text{pipe}}$, as defined in (17), or 0, otherwise. The vector of all ones is denoted by $\mathbb{1}$.

The equations for flow conservation at the nodes and pressure drop on the arcs read:

$$\mathcal{A}\boldsymbol{Q_0} = \boldsymbol{d}, \tag{33}$$

$$\boldsymbol{\Delta} - \mathcal{A}^T \boldsymbol{\pi} - \alpha \, |\boldsymbol{Q_0}| \, \boldsymbol{Q_0} = -\pi_{\overline{u}} \, \mathcal{A}^T \mathbb{1}, \tag{34}$$

where (34) is built from the pressure loss equations (15) for pipes by neglecting geodetic heights and adding the vector $\boldsymbol{\Delta}$ for pressure changes in compressor groups or control valves. The compressibility factor z is approximated by (5). Splitting $\mathcal{A} = (\mathcal{A}_B, \mathcal{A}_N)$ into basis and non-basis parts according to a spanning tree, (33)–(34) are equivalent to

$$\boldsymbol{Q_{0,B}} = {\mathcal{A}_B}^{-1}\boldsymbol{d} - {\mathcal{A}_B}^{-1}\mathcal{A}_N \boldsymbol{Q_{0,N}}, \tag{35}$$

$$\boldsymbol{\pi} = \pi_{\overline{u}} \, \mathbb{1} - ({\mathcal{A}_B}^T)^{-1}\big(\alpha_B \, |\boldsymbol{Q_{0,B}}| \, \boldsymbol{Q_{0,B}} - \boldsymbol{\Delta_B}\big), \tag{36}$$

$$\alpha_N \, |\boldsymbol{Q_{0,N}}| \, \boldsymbol{Q_{0,N}} - \boldsymbol{\Delta_N} = {\mathcal{A}_N}^T ({\mathcal{A}_B}^T)^{-1}\big(\alpha_B \, |\boldsymbol{Q_{0,B}}| \, \boldsymbol{Q_{0,B}} - \boldsymbol{\Delta_B}\big). \tag{37}$$

Indeed, inserting (35) into (34) yields

$$\boldsymbol{\Delta_B} - {\mathcal{A}_B}^T \boldsymbol{\pi} - \alpha_B \, |\boldsymbol{Q_{0,B}}| \, \boldsymbol{Q_{0,B}} = \; -\pi_{\overline{u}} \, {\mathcal{A}_B}^T \mathbb{1}, \tag{38}$$

$$\boldsymbol{\Delta_N} - {\mathcal{A}_N}^T \boldsymbol{\pi} - \alpha_N |\boldsymbol{Q_{0,N}}| \boldsymbol{Q_{0,N}} = \; -\pi_{\overline{u}} \, {\mathcal{A}_N}^T \mathbb{1}. \tag{39}$$

Multiplying (38) with $-({\mathcal{A}_B}^T)^{-1}$ implies (36), and inserting (36) into (39) yields (37).

Let us add some remarks on how these formulae relate to network issues:

- The arcs corresponding to the columns of $\mathcal{A}_B$ form a spanning tree $\mathcal{T}_B \in \bar{G}$.
- Each arc a belonging to a column of $\mathcal{A}_N$ stands for a fundamental cycle with respect to $\mathcal{T}_B$, i.e., the unique cycle in $\mathcal{T}_B \cup a$.
- The columns of ${\mathcal{A}_B}^{-1}$ mark the directed paths in the spanning tree from the root node to all other nodes, with entry $+1$ (-1) if the arc is directed in the same (opposite) way as the path from the root, and with 0 if the arc does not belong to the path.
- The columns of ${\mathcal{A}_N}^T({\mathcal{A}_B}^T)^{-1}$ mark the tree arcs in fundamental cycles. The column corresponding to arc a is the difference of the columns of the paths from the root to the head of a and that to its tail.
- The representation of $\boldsymbol{\pi}$ in (36) states that the (squared) pressure at an arbitrary node $u \neq \overline{u}$ is determined by the pressure at the root minus the pressure drop along the unique path from $\overline{u}$ to u and plus/minus the pressure changes by the active arcs along the path.
- The identity (37) states that the pressure change along the tree arcs of each fundamental cycle must equal the change along the arc that created the cycle.

Compared to (33)–(34), one ends up with a much smaller implicit part in the transformed system, namely (37). It has just as many equations as there are components of

$\boldsymbol{Q_{0,N}}$ or as there are fundamental cycles in $\bar{G}$. For strongly meshed gas distribution networks, such as those met in many parts of Europe, this number still can be substantial. For weakly meshed gas transportation networks, however, values of 1 or 2 already become practically relevant. If the cycles of $\bar{G}$ are arc disjoint, then (37) is separable with respect to the components of $\boldsymbol{Q_{0,N}}$.

Altogether, the feasibility system we use comprises (37), pressure bounds applied to the right-hand sides of (36), and the following specific conditions for resistors, control valves, and compressor groups.

Control valves allow for a pressure reduction, when traversed in the nominal direction. Compressor groups are modeled at an aggregated level, without resolution to individual machines. We add the following restrictions for $a = (u, v) \in A_{\mathrm{cg}}$:

$$p_u \geq p_v, \qquad (p_u - p_v) \cdot (-Q_{0,a}) \leq 0 \tag{40}$$

$$(p_u - p_v) \cdot (\underline{Q}_{0,a} - Q_{0,a}) \leq 0, \qquad (p_u - p_v) \cdot (Q_{0,a} - \overline{Q}_{0,a}) \leq 0 \tag{41}$$

$$(p_u - p_v) \cdot \left(\underline{rat}_a - \frac{p_v}{p_u}\right) \leq 0, \qquad (p_u - p_v) \cdot \left(\frac{p_v}{p_u} - \overline{rat}_a\right) \leq 0 \tag{42}$$

$$(p_u - p_v) \cdot (\underline{inc}_a - p_v + p_u) \leq 0, \qquad (p_u - p_v) \cdot (p_v + p_u - \overline{inc}_a) \leq 0. \tag{43}$$

Inequalities (40) specify that compressor groups allow for a pressure increase in their active mode when traversed in the nominal direction. In order to approximate the characteristic diagram, lower and upper bounds ($\underline{Q}_{0,a}$ and $\overline{Q}_{0,a}$, respectively) are imposed on the flow for the active mode (see (41)). We impose lower and upper bounds ($\underline{rat}_a$ and $\overline{rat}_a$, respectively) on the pressure ratio (see (42)) and on the pressure increase ($\underline{inc}_a$ and $\overline{inc}_a$, see (43)). Resistors are modeled according to their type by the respective equation (13) or (14) in a non-smooth way due to the occurrence of the sign of the flow rate.

Search for Promising Switching Decisions To enable the approach described above, we have to fix switching states for control valves, valves, and compressor groups. We use two different techniques to fix these binary decisions. First we use a transshipment heuristic, and then we try some sets of given switching states for all active elements. These sets are constructed by using expert knowledge about the network and by collecting sets of switching states from the transshipment heuristic in cases where they led to a feasible solution for some nominations.

Building the Transshipment Model In gas networks often there are subnetworks of elements representing bigger entities typically comprising compressor groups and other active arcs, enabling a limited number of internal flow paths only. We collapse internal nodes and arcs of such an entity into a single node, which is connected to all nodes on the boundary of the entity in both directions. The arcs are assigned small cost coefficients. The node representing the entity is assigned the balance of all in- and outflows related to nodes within the entity.

All switchable arcs, i.e., valves, control valves, and compressor groups, outside of entities are substituted by one or two directed arcs depending on the signs of the flow bounds $\underline{Q}_{0,a}$ and $\overline{Q}_{0,a}$. Cost coefficients are assigned heuristically, for instance, in increasing order starting with forward arcs representing compressor groups, followed by forward arcs modeling control valves and arcs representing valves pointing into both directions. Finally, backward arcs for control valves and compressor groups are assigned the largest costs.

Each connected component of the network remaining after the removal of all entities and switchable arcs is collapsed into a single node, which is assigned the balance of the nominations for the entries and exits within the component. Arcs in both directions connect the node representing such a connected component with all boundary nodes of entities and endpoints of other switchable elements being incident to the component. To all these arcs, the sum of the friction coefficients α_a in the component is assigned as cost.

Using the Transshipment Model The moderate size of the transshipment problem allows for rapid solution by any state-of-the-art linear programming solver. From the optimal solution, switching states for active arcs in the original network are derived. Namely, if in the optimal solution there is no flow passing through an element outside an entity, we choose the off-state for it. Reflecting expert knowledge, for all the elements inside a specified entity, a suitable decision from the corresponding subnetwork operation mode is chosen by a set of rules based on the amount of flow and the flow direction through the entity.

3.4 The MPEC Approach

As seen in Section 2, the problem of validation of nominations is a discrete-continuous nonlinear and non-smooth feasibility problem. The approach described here aims at finding feasible solutions by means of NLP techniques. Since these techniques require a continuous and sufficiently smooth model (C^2 in our case), we reformulate the discrete and non-smooth aspects in an appropriate way. For a more detailed explanation of the theory behind this approach see [67, 69].

Smoothing Techniques Many physical and technical aspects of NoVa are non-smooth, such as the pressure loss at pipes or resistors. Most of them can be handled by standard smoothing techniques for min, max, sgn, and absolute value functions (cf. [68]), but others require problem-tailored smoothing techniques. As an example, we present the smoothing of our non-smooth model of the pressure loss in pipes, which is based on the quadratic pressure loss model (15). For the compressibility factor we choose the AGA formula (5), see, e.g., [68]. Obviously, the term $|q_a|\, q_a$ in (15) has a second-order discontinuity at zero, while the composite friction model $\lambda^{\mathrm{HPPC}}(q_a)$ defined by (6), (7) has a jump discontinuity at the transition from laminar to turbulent flow, see Figure 4 (left). Both non-smooth aspects are handled by a global smooth approximation developed in [12, 11, 68]:

$$\phi(q_a) \approx \lambda^{\mathrm{HPPC}}(q_a)\, |q_a|\, q_a. \tag{44}$$

Our choice of $\phi(q_a)$ is asymptotically correct for $|q_a| \to \infty$; see Figure 4 (right).

Complementarity Constraints for Combinatorial Aspects Active network elements like compressor groups introduce discrete decisions into the validation problem. For a compressor group $a = (u, v) \in A_{\mathrm{cg}}$, these discrete decisions result in states like *active* or *closed* that can be described in a simplified way by

$$\begin{aligned} a \text{ is active/in bypass} \quad &\Longrightarrow \quad p_v - p_u - \Delta_a = 0, \quad \Delta_a \geq 0, \\ a \text{ is closed} \quad &\Longrightarrow \quad q_a = 0, \end{aligned} \tag{45}$$

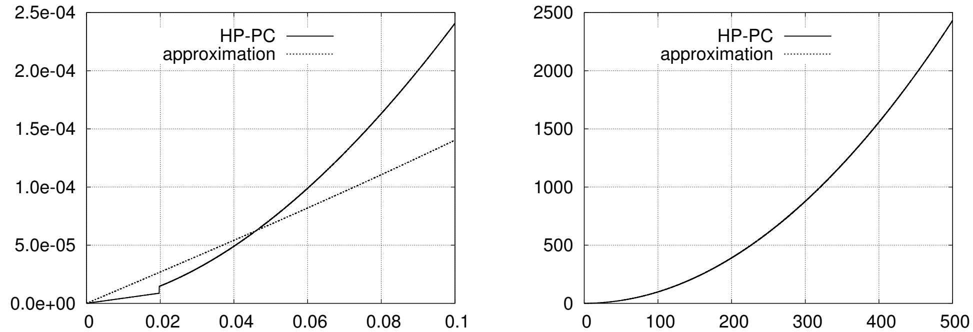


Figure 4: Discontinuous friction model λ^{HPPC} and its smooth approximation ϕ for small flow rates (left) and large flow rates (right)

with Δ_a denoting the pressure increase. Standard mixed-integer approaches for modeling (45) use binary variables for the states (cf. Section 3.1 and 3.2). Here we follow a different approach and replace (45) by the *complementarity constraint*

$$\tilde{p}_a\, q_a = 0, \quad \Delta_a \geq 0, \quad \tilde{p}_a := p_v - p_u - \Delta_a. \tag{46}$$

The model for the active state in (45) is not sufficiently detailed to be practically relevant. An active group can actually be operated in several configurations $k \in \mathcal{K}_a$, each configuration being a serial arrangement or parallel combinations of compressors. Our heuristic approach attempts to determine the "most feasible" configuration for a given flow-pressure-situation (p_u, p_v, q_a). To this end, we make the fictitious assumption that the entire gas flow passes through every configuration, $q_a = q_a^k$ for all $k \in \mathcal{K}_a$, and relax the pressure increase $\Delta_a = p_v - p_u$ in (45) to a convex combination of pressure increases of the individual configurations, $\Delta_a^k = p_v^k - p_u^k$. This is completed by a feasibility model $(p_u^k, p_v^k, q_a^k, \mathcal{S}_a^k) \in \mathcal{F}_a^k$ for every $k \in \mathcal{K}_a$, consisting of a set of smooth nonlinear equations and inequalities and a set of slack variables $\mathcal{S}_a^k$; see [68] for details. The compressor group model thus takes the form

$$\begin{aligned} \forall k \in \mathcal{K}_a: \quad \Delta_a &= \sum_{k\in\mathcal{K}_a} \sigma_a^k \Delta_a^k, \quad \sum_{k\in\mathcal{K}_a} \sigma_a^k = 1, \quad \sigma_a^k \geq 0, \\ q_a &= q_a^k, \quad (p_u^k, p_v^k, q_a^k, \mathcal{S}_a^k) \in \mathcal{F}_a^k. \end{aligned} \tag{47}$$

To obtain the "most feasible" configuration with the refined model (47), we minimize a suitable norm of the slack variables. Then we can heuristically choose the configuration k with a minimum slack variable value and a maximum coefficient σ_a^k.

Using techniques similar to (46) for the remaining active elements, almost all discrete aspects can be represented by complementarity constraints. In combination with the smoothing techniques, the NoVa problem is thus transformed into a *mathematical program with equilibrium constraints* (MPEC). We apply standard MPEC regularization schemes as in [70, 28, 41] to obtain a smooth and continuous NLP reformulation.

A Two-Stage Solution Approach The NLP obtained from the MPEC approach combines highly nonlinear non-convex physical and technical phenomena with numerically problematic smoothing and penalization techniques. Numerical experiments show that solving all these aspects simultaneously on real-world networks is hardly possible with general purpose NLP codes like IPOPT [80] or SNOPT [36]. As a substantially more robust and reliable solution procedure, we propose a two-stage approach, where each stage solves an NLP with a different set of model aspects.

The first stage incorporates all previously described features except for the convexification of compressor groups (47). Thus, it decides the principal states (open or closed) of all active elements.

As mentioned above, various regularization schemes exist for the stage 1 MPEC. In our numerical experiments the penalization scheme of [28] proved to be the most appropriate choice for the specific class of problems. This scheme introduces an NLP sequence NLP(τ_k), $k = 1, 2, \ldots$, with decreasing penalization parameter τ_k. While providing convergence theory, the approach suffers from significant practical drawbacks: the entire sequence needs to be solved to optimality, and the computation time is a multiple of a single instance. In our approach, τ is instead handled as an optimization variable that is driven to zero during the NLP solution process by additional constraints or penalization schemes. This *direct* approach is very stable for our practical purposes.

The solution of the first stage is analyzed and translated into discrete decisions. An ambiguous situation arises when one or more complementarity constraints are satisfied bi-actively, with both factors equal to zero. In this case, the most promising decision is chosen heuristically for stage 2, based on empirical experience. After the first stage, all open/closed states are decided and the overall flow situation in the network is determined; these will be fixed in the second stage.

In the second stage, the active configurations of the compressor groups are to be determined. The decisions of stage 1 are fixed, and the convexification model (47) is added. Moreover, we use the solution of stage 1 to initialize the NLP of stage 2.

Then, the solution of the second stage is analyzed to fix the remaining active configurations. If more than one configuration with vanishing slacks exist, the one with the largest convex coefficient is chosen. If no configuration with vanishing slack norm exists, the configuration with the smallest constraint violation is used.

3.5 Validation by NLP

Since the full NoVa problem becomes intractable if both a detailed physics model and discrete decisions are incorporated, each of the four solution approaches employs its specific approximations of certain physical and technical details. This raises the need for a posteriori feasibility checks with respect to some reference model that is trusted to provide a sufficiently accurate description of reality, such as the models used in commercial gas network simulation software.

We use a model that includes the pressure loss equation (15), with a global smooth approximation, see, e.g., [12, 68], replacing the piecewise friction model (6) and (7); Compressor groups are modeled as accurately as possible, complete with drives, with operation ranges of individual units, and with arbitrary distributions of flow among parallel units, see [68]; only the fuel gas is neglected in the flow balances, and gas temperatures as well as gas quality parameters are considered constant in the entire NLP model; details can be found in [68]. Valves and control valves are modeled as in (9) and (10), and both resistor types of [68] arise.

The difficulty is that solution candidates with approximate physics from any approach will generically be infeasible in a strict sense; we can only expect approximate feasibility. Assuming that the discrete variables of a solution candidate are correct in the sense that suitable "small" modifications of the continuous variables will yield exact feasibility, we proceed as follows to obtain a high-accuracy feasible solution: we fix all discrete decisions and all discrete states based on the given solution candidate. The discrete-continuous problem with a detailed physics model then reduces to a purely continuous feasibility problem consisting of linear and nonlinear equalities and inequalities with suitable smoothness properties (C^2 in our case). We introduce slack variables to relax all the nonlinear constraints. The minimization of some measures of the total constraints violation, specifically a weighted ℓ_1-Norm of the slacks (with large weights on the slacks of compressor units), then yields a standard NLP. An initial solution estimate for this NLP is generated from the given solution candidate. If we are successful in computing a local NLP minimizer whose slack objective is zero or sufficiently small, we have a complete solution of the original problem, and we will regard the given candidate as a *valid* approximate solution. The final NLP solution can ultimately be verified with a suitable simulation tool.

Note that a different outcome of our NLP validation procedure does not provide any decisive information on the original problem. If a local minimizer with a nonzero slack objective is computed, we know that one or more constraints of the original problem are violated. In case of a "small" objective, further (typically manual) checks may be carried out to decide whether the minimizer is practically acceptable or not. In case of a "large" objective, we just know that the given solution candidate did not lead to a solution with zero slack. If the instance is feasible, one possibility is to improve the candidate by increasing the modeling accuracy in the first-stage approach.

4 Computational Studies

To show the practical relevance of our approaches as a solver for the NoVa problem in gas networks, they are applied to two different types of nominations for country-size real-world gas networks arising at our project partner Open Grid Europe (OGE), see Figure 5 for an illustration. The first set (*SN4*) contains 4227 automatically generated nominations based on contractual and statistical data by a sampling approach, see [45]. All these nominations are based on the same network with 592 nodes, 425 pipes, 35 valves, 23 control valves, and 6 compressor groups. The second group (*AB6*) consists of 44 hand-made worst-case nominations by OGE, including four definitely infeasible instances. The corresponding networks are variations of the above. They have about 660 nodes, 500 pipes and more than 30 valves, 25 control valves, and 7 compressor groups. In the appendix we give detailed results for this second test set, including feasibility status.

4.1 Solutions via Black-Box Solvers

Since NoVa is a mixed-integer nonlinear problem, it is a fundamental question whether it is possible to solve our test instances with state-of-the-art MINLP solvers. To answer this question, we performed a computational study. The answer clearly depends on the mathematical model that is used to represent NoVa. We choose a MINLP model based

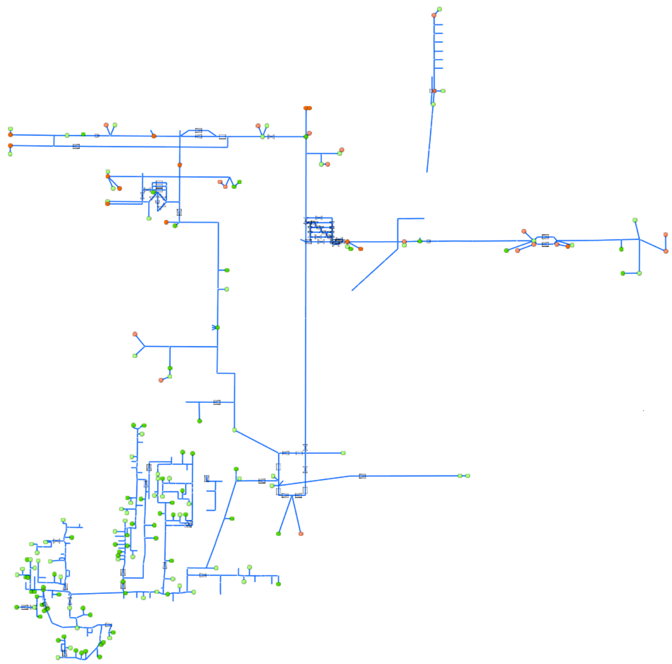


Figure 5: Illustration of one network used for the computations

on (15) with the simplifications (20)–(21), (11), and (12). Moreover, we include the non-convex characteristic diagrams of compressors; the remaining model parts are very close to those of MILP approach.

We are interested in how state-of-the-art solvers perform on such MINLP instances. If a local MINLP solver finds a feasible solution, NoVa is solved affirmatively. If a local solver is, however, not able to find a feasible solution, no conclusion for NoVa can be drawn. Thus, to prove infeasibility of a nomination—a crucial point for our application—the usage of a global solver is required. We apply state-of-the-art solvers of both classes: As global solvers for non-convex MINLPs, we select BARON [77, 76, 75] and SCIP [1, 78]. These solvers mainly implement (linear) convexification techniques in combination with a spatial branch-and-bound.

As local MINLP solvers, we use BONMIN [7], ALPHAECP [81], and KNITRO [13]. All three solvers are exact for convex MINLPs, but can be used as heuristics in the non-convex case. BONMIN implements different algorithms for convex MINLP, from which we choose two different variants, BB and Hyb, for our studies. Variant (BB) implements a nonlinear branch-and-bound search based on solving continuous nonlinear relaxations at the nodes. The hybrid algorithm (Hyb) is a combination of an outer approximation branch-and-cut algorithm and the BB algorithm. ALPHAECP is developed for solving convex or pseudo convex MINLPs by solving a sequence of MILPs, occasionally solving NLP subproblems, and generating cutting planes. From KNITRO we select the nonlinear branch-and-bound method.

All computations in this subsection were performed sequentially using a single thread on a machine with two six-core AMD Opteron CPUs with 2.6 GHz and 64 GB of RAM.

We use GAMS 23.8.2 [32] to communicate over a common interface with all solvers. The respective solver versions included in this GAMS release are BARON 10.1, SCIP 2.1.1, BONMIN 1.5, ALPHAECP 2.09.01, and KNITRO 8.0. All solvers are set to use their default parameters (if not stated otherwise above). Experiments with other parameters did not significantly improve the results. In addition, we use the nonlinear bound propagation preprocessing (except for SCIP), which is also used to tighten variable bounds for the MILP approach (see Section 4.3 for further details). For SCIP, this preprocessing dramatically worsens the results and sometimes even leads to false infeasibility detections. The other solvers did not behave in this way.

From the first test set (SN4), we pick a random subset of 50 instances to keep the computational effort within reasonable bounds. We set a time limit of four hours for each solver and instance, which is twice the time limit that is subsequently used for our specialized approaches. BARON solved three out of 50 instances by finding a feasible solution after 65, 112, and 162 minutes within the time limit. SCIP solved two instances in 3 and 10 minutes and ran into the time limit for all others. BONMIN, ALPHAECP, and KNITRO are unable to find any feasible solution.

For the second test set (AB6), BARON solved three out of the 44 instances by finding a feasible solution after 84, 112, and 219 minutes. SCIP solved two instances in 1 and 63 minutes. Again all local solvers failed to find a feasible solution.

Interestingly, no global solver is able to prove infeasibility, although 12 out of the 50 instances in the first test set (SN4) are infeasible and at least four instances in the second test set (AB6).

We conclude from these computational experiments that large-scale instances of the NoVa problem cannot be solved with state-of-the-art black-box solvers. This motivates the approaches discussed in this paper.

4.2 Computational Setup

In the following we discuss the computational results of the four approaches presented in Section 3 and the corresponding validation step. The computations were performed on a Linux cluster. Each node has two Xeon 3.2 GHz quad core processors and 48 GB of RAM. We imposed a time limit of two hours for the application of the approaches introduced in Section 3. We performed single-threaded computations, except for Section 4.3. On each node, only one job is executed simultaneously. The run times for the four approaches to find good discrete decisions exclude the timings for NLP validation. All timings include the time for reading the data and building the model, which for some instances consumes a major part of the running time.

Gurobi 5.0 [38] was used to solve the constructed problems by means of the MILP approach. The SB approach was implemented in a prerelease version of SCIP 3.0, see [1, 78, 71]. The LP and NLP subproblems therein were solved using CPLEX 12.4 [42] and IPOPT 3.10 [80], respectively. IPOPT 3.10 was also used to solve the NLP problems in the RedNLP and MPEC approaches. Since the validation NLP can be tackled by several NLP solvers, we sequentially tried the solvers IPOPT 3.10, CONOPT 3.15C, CONOPT 4.00 [23], and KNITRO 8.0.0 [13] until one of them converges to a feasible point. This last step could, of course, be parallelized. The NLPs of the MPEC and RedNLP approach as well as the NLP validation are solved using GAMS [32] version 23.8.2 as an interface.

Most of the results are illustrated by performance diagrams, where at each point on the x-axis (corresponding to some given measure, e.g., running time, slack value, etc.) we display on the y-axis the fraction of the total number of instances that were solved using at most the given measure on the x-axis. All times are reported in seconds.

4.3 Solutions of the MILP Approach

In this section we discuss numerical results of the MILP approach described in Section 3.1. All MILPs are solved with the branch-and-cut solver Gurobi with default parameter settings on 8 threads, except that dual reductions and precrush are disabled, since we apply (lazy) cutting planes.

Before the MILP model is constructed, a straightforward nonlinear bound propagation preprocessing is performed to improve unnecessarily large variable bounds. Since the sizes of the variable domains have a direct impact on the size of the linearization, this step is crucial. We refer to [33] for a detailed description of this propagation algorithm. The preprocessing never took more than 10 seconds and is therefore negligible compared to the overall running time of this approach. Thus, we do not list presolving times explicitly. The piecewise linear relaxations are constructed to be within a deviation of at most 1.5 bar from the underlying nonlinear function. We remark that if this tolerance is relaxed, the resulting MILPs are solved faster, but the number of validated solutions decreases. Conversely, when the error tolerance is strengthened, the amount of nonzero slack validations declines, while the running time increases. The error tolerance value of 1.5 bar is an appropriate compromise based on our experience with different test sets.

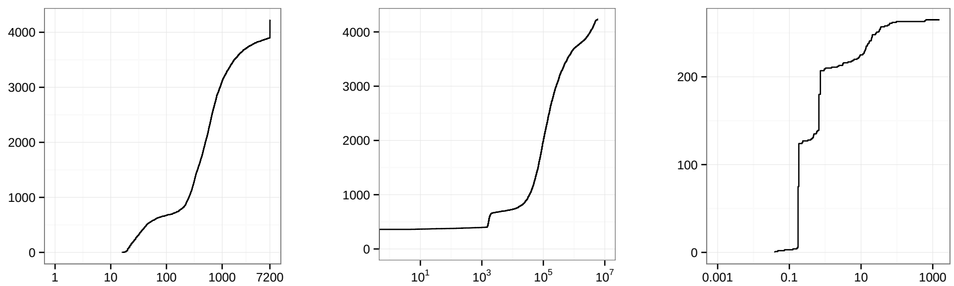
An illustration of the results for the first test set (SN4), consisting of 4227 instances, is depicted in Figure 6. The variation of the sizes of the resulting MILP instances is small. On average the instances have about 18302 constraints and 11222 variables, about 3833 are binary variables.

For 3510 instances, a feasible solution to the MILP model is found, and 694 instances are proved to be infeasible during the time limit of two hours. This leaves 23 undecided instances. Gurobi's running time is presented in Figure 6(a). The average running time is about 20 minutes, and the median is about 9 minutes. A subset of 3205 instances was solved to optimality, and 3444 instances are proved to be within 10 % of the optimum. Of the 3510 feasible instances, the NLP validation confirmed 3245 of them, i.e., resulted in a zero slack value. Figure 6(c) shows the distribution of the slack sum values for the NLP validation of the remaining 265 instances.

The average values are mainly influenced by the large time limit. Within a stricter time limit of 20 minutes, for example, we still find feasible solutions for 3285 instances. A similar behavior can be seen for the number of branch-and-bound nodes shown in Figure 6(b).

Almost all instances (389 out of 397) that were solved within less than 1000 branch-and-bound nodes (or within less than approximately 30 seconds) are infeasible instances. In general, only 40 of the 694 infeasible instances needed more than 10000 nodes or more than approximately one minute. We conclude that the MILP model is suitable for detecting infeasibility, one of the main purposes why the model is constructed as a relaxation of the underlying nonlinear model.

We illustrate a typical solution process of the test set (SN4) on instance number 1872. The solution of this instance requires about 20 minutes. The first feasible solution is



(a) Running times (s)

(b) Number of branch-and-bound nodes

(c) NLP slack sums for 265 MILP feasible instances with positive slack

Figure 6: Numerical results on the first test set (SN4) for the MILP approach.

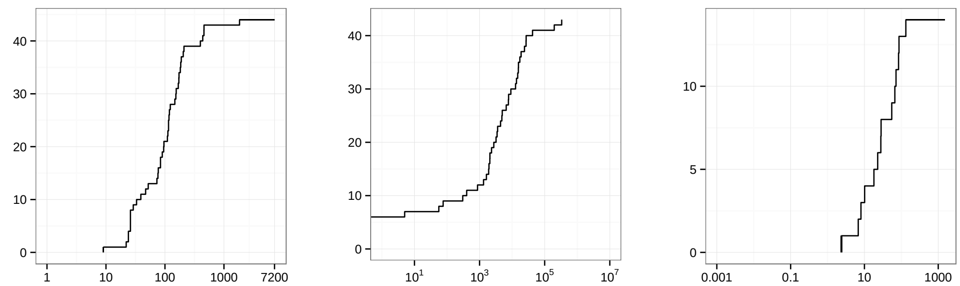
found after 214 seconds and 38851 branch-and-bound nodes, yielding an optimality gap of 98 %. The dual bound is typically weak at the beginning, but this bound is significantly improved up to the point where the initial solution is found. Subsequently, the optimality gap is constantly reduced to 0.24 % (by both improving the primal and dual bound) until the optimal solution is found after 615 seconds have passed and 104874 nodes have been explored. The remaining search process, which is another 585 seconds and 379938 nodes, is spent to prove optimality.

The average solution time can in principle be halved, if the solution process is stopped as soon as the gap between the primal and dual bound is less than 1 %. We recall in this context that NoVa is a feasibility problem. The objective in the MILP approach has been added to overcome the gap between the underlying nonlinear model of the MILP approach and the physically more detailed nonlinear validation model. An optimality gap limit of 1 % would also have a major impact on instances reaching the time limit. Most of those instances (188 out of 305) in fact have a similar behavior: a first feasible solution is found within the first 5 minutes and is improved to a gap of less than 1 % within an overall running time of about 10 minutes. The remaining time until the time limit is then just used for trying to prove the last per cent of optimality.

The results for the second test set (AB6) consisting of 44 expert nominations are shown in Figure 7. The MILP instances have on average about 23824 constraints and 14871 variables, 5484 of which are binary.

For 39 instances, a feasible solution to the MILP model is found, and five instances are proved to be infeasible. The infeasibility of one instance is already detected by the nonlinear bound propagation preprocessing, while the remaining four infeasibilities are determined by Gurobi. Each instance is solved within a running time of no more than 30 minutes. The running times are depicted in Figure 7(a). The average running time is about 5 minutes, and the median is about 3 minutes. From the 39 feasible instances, 25 were confirmed with a zero slack by the NLP validation.

A more detailed view into the solution process of these instances shows that infeasibility is again rapidly detected (less than a few seconds). In contrast to the first test set, the dual bound is already improved in the root node or at least within the first 100 nodes of the search tree. A first primal solution is typically found faster, too – on average after

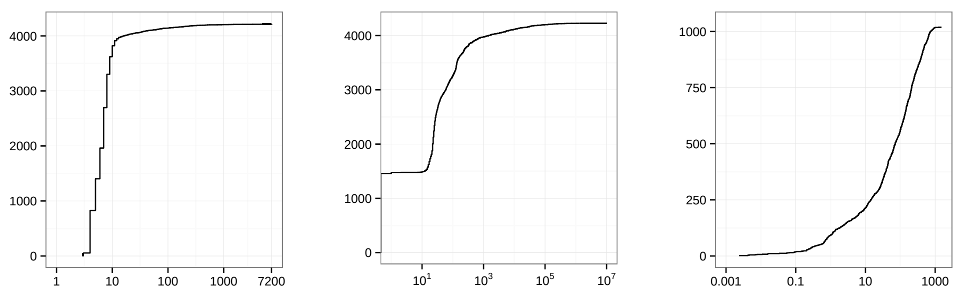


(a) Running times (s)

(b) Number of branch-and-bound nodes

(c) NLP slack sums for 14 MILP feasible instances with positive slack

Figure 7: Numerical results on the second test set (AB6) for the MILP approach.



(a) Running times (s)

(b) Number of branch-and-bound nodes

(c) NLP slack sums for 1036 SB feasible instances with positive slack

Figure 8: Numerical results on the first test set (SN4) for the SB approach

approximately 100 seconds and exploring about 2000 nodes. Another observation is that the optimality proof in general does not require as much effort as in the first test set. Typically, just a few more nodes are explored.

An explanation for the remarkable difference between the solution process of instances of the test set (SN4) and (AB6) might be their different origins. The nominations of (SN4) are based on ordinary, everyday gas delivery situations, which are typically realizable in many different ways and thus typically contain multiple feasible (nearly) symmetrical solutions. The nominations from (AB6), however, are based on expert knowledge to describe exceptional extreme situations, in which the number of admissible discrete decisions is much lower.

4.4 Solutions of the SB Approach

In this section, numerical results for the SB approach (see Section 3.2) are presented. On the SN4 test set, 4213 instances are solved, leaving only 14 without solution or proof of infeasibility. 600 instances are proved to be infeasible within the model used by this approach. For 3613 instances, a feasible solution is found within the time limit. Figure 8 shows the performance profiles for the run time, branch-and-bound nodes, and NLP validation slack values of this solver on the SN4 test set. The profile shows that the approach solves 90 % (3270) of the instances within less than 10 seconds. More than 97 % of the instances are solved within one minute.

Most solutions (2640 out of 3613) were found in the tree by the *subnlp* heuristic, which is included in SCIP by default. Whenever a solution is found that is integer feasible for the linear relaxation (either by solving the relaxation in a node or by applying a MILP heuristic), the subnlp heuristic applies a local solver (IPOPT) to the NLP that is obtained from the MINLP by fixing all integer variables. In our application, this proves to be extremely effective.

The picture looks similar for the 600 instances for which infeasibility could be proved. Here, in 484 of the instances SCIP presolving already detected the infeasibility. Overall the running time was less than 10 seconds for 92 % of the 600 instances.

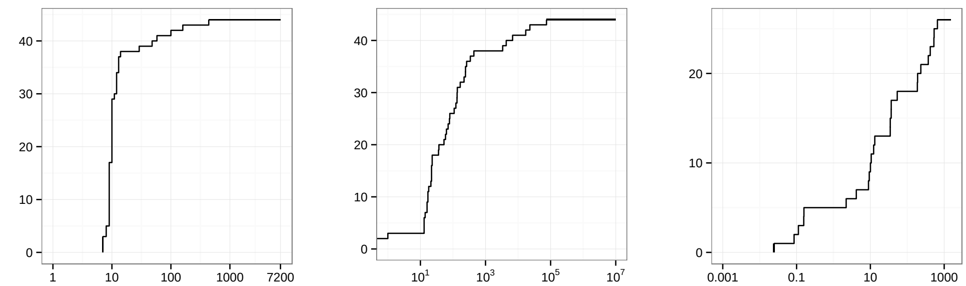
While the SB seems extremely fast on a large part of the test set, there are 14 instances for which neither a solution could be found nor infeasibility could be proved within the time limit of two hours. Two main reasons can be identified. For two instances, integer feasibility of the relaxation is hard to reach. In one of these two instances, no integer feasible relaxation is found at all during the solution process. This instance can be proved to be infeasible with slightly different settings of the solver. The remaining unsolved instances spend more than half of the time trying to find a feasible solution in the subnlp heuristic. On these instances, substantial effort is made to strengthen the relaxation by spatial branching; between 30 and 98 % of the branchings are performed on continuous variables. However, with customized settings, all unsolved instances can be solved within two hours resulting in 12 feasible and two infeasible instances.

On the vast majority of the instances, surprisingly small effort is made to strengthen the relaxation by spatial branching. Only in 133 instances, spatial branching is applied at all. On the remaining test set, the relaxation is strengthened by cutting planes, but branching is not needed. Since we first branch on integer variables that have fractional values in the relaxation, this can be expected for the instances that are solved with only a few nodes. However, only 12 of the 100 instances that take most time to find a feasible solution apply spatial branching. The solution of the most time consuming feasible instance (4649 seconds), for example, does not perform any spatial branching. In this instance, 719290 nodes are explored of which only three have an integer feasible solution of the linear relaxation. In the third integer feasible node, the subnlp heuristic finds a solution.

The SB approach was able to solve all instances from the AB6 test set. Four instances were recognized to be infeasible, while for the remaining 40 instances, feasible solutions could be computed. Performance profiles for this test set are depicted in Figure 9.

For two instances, infeasibility was proved in presolving. Proving infeasibility for the other two instances took 436 and 159 seconds, respectively. Interestingly, neither instance was solved using spatial branching. All feasible solutions were found in less than 100 seconds, but only 65 % in less than 10 seconds.

The analysis of the validation of the solutions produced by this approach is as follows. On the SN4 test set, 2577 out of 3613 (71 %) solutions could be validated with slack zero, while for 1036 a positive slack remains in the NLP validation. On the AB6 test set, only 14 out of 40 solutions (35 %) validate with slack zero. Since only one (almost randomly



(a) Running times (s)

(b) Number of branch-and-bound nodes

(c) NLP slack sums for 26 SB feasible instances with positive slack

Figure 9: Numerical results on the second test set (AB6) for the SB approach

chosen) solution is passed as a candidate to the validation procedure, local improvement heuristics after a failed validation could help to increase the number of validated solutions.

4.5 Solutions of the RedNLP Approach

For each nomination, the RedNLP approach first uses the binary decisions from the transshipment problem. If this is not successful, the algorithm is continued by testing 33 given configurations, which successfully solved other nominations. The solution of the reduced NLP model is given as a starting point to the detailed validation NLP. The solution is called *confirmed* if a solution with zero slack could be found.

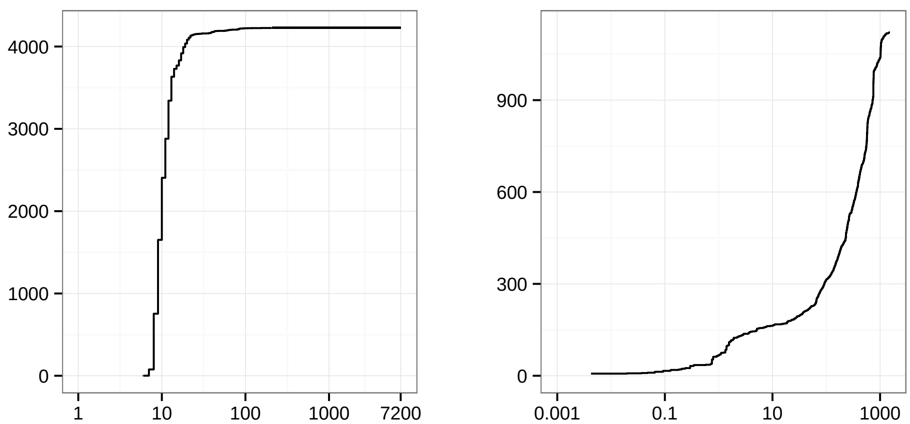
For the first test set (SN4) of 4227 nominations generated from statistical information, the reduced NLP model can solve 4194 nominations, 3731 of which were found using the switching decisions derived from the solution of the transshipment problem. 3058 of the solutions were confirmed by the detailed NLP slack model. The total computing times for the reduced NLP range from 6 to 208 seconds with an average of 11.78 seconds and a median of 10 seconds. Figure 10 shows the results for this test set.

Out of the 44 nominations in the second test set (AB6), a solution was found for 39 nominations, 27 of which were confirmed. 22 of the solutions were found based on the transshipment solution. The computing times range from 12 to 101 seconds with an average of 31.64 seconds and a median of 18.5 seconds. Figure 11 presents the results.

The transshipment problem for this network has 198 variables and 108 equality constraints. The size of the reduced NLP depends on the chosen switching decisions. The number of variables varies between 1408 and 1441, the number of equality constraints between 1318 and 1346, and that of inequality constraints between 60 and 85.

4.6 Solutions of the MPEC Approach

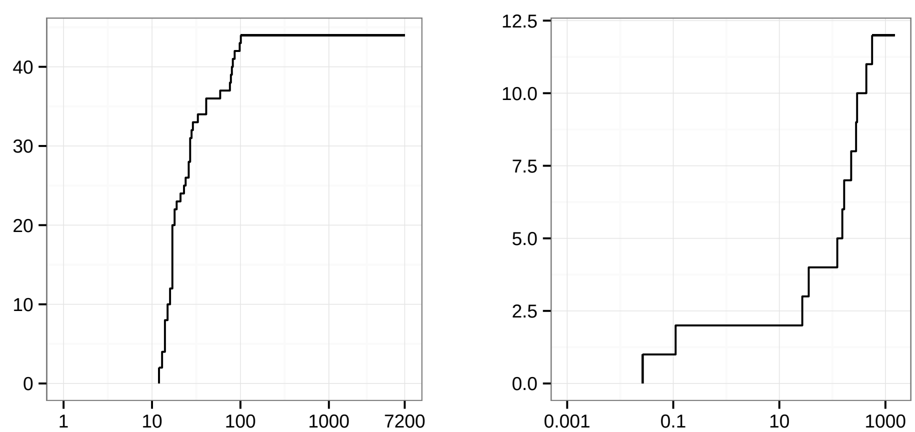
The results of the MPEC approach are given in Figures 12 and 13. The 4227 instances of the first test set (SN4) require an average computing time of about 17 seconds, with a maximum of 89 seconds. 2927 of these instances are solved successfully. The final NLP stage requires about 7 seconds on average with a maximum of 42 seconds, and



(a) Running times (s)

(b) NLP slack sums for 1136 RedNLP feasible instances with positive slack

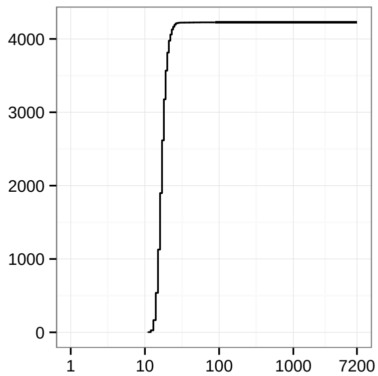
Figure 10: Numerical results on the first test set (SN4) for the RedNLP approach



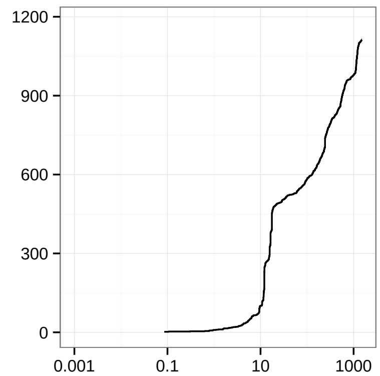
(a) Running times (s)

(b) NLP slack sums for 12 RedNLP feasible instances with positive slack

Figure 11: Numerical results on the second test set (AB6) for the RedNLP approach

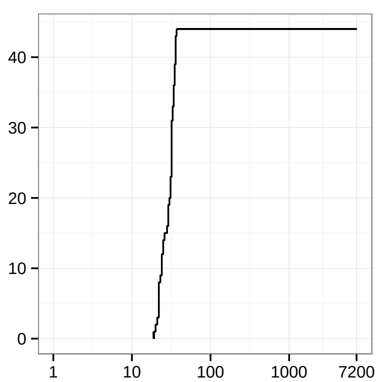


(a) Running times (s)

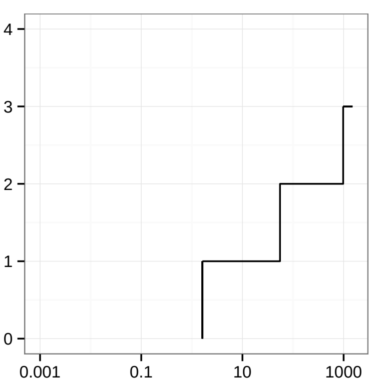


(b) NLP slack sums for 1178 MPEC feasible instances with positive slack

Figure 12: Numerical results on the second test set (SN4) for the MPEC approach



(a) Running times (s)



(b) NLP slack sums for 4 MPEC feasible instances with positive slack

Figure 13: Numerical results on the second test set (AB6) for the MPEC approach.

1749 of the MPEC-feasible instances are solved to optimality with vanishing slacks. All computing times include a preprocessing of variable bounds, which only takes a few seconds, see Section 4.3.

Constraints that couple discrete decisions of different active elements to select feasible subnetwork operation modes (see Section 2) can currently not be handled within the MPEC model. These constraints do not fit the requirements necessary for the problem-tailored MPEC-based reformulation techniques, that are used for the other discrete aspects of the model. Only 30 of the 2927 instances solved by the validation NLP satisfy these additional constraints. The success rate may be increased by postprocessing algorithms which revisit ambiguous decisions. These algorithms are subject of further research.

For the 44 nominations in the second test set (AB6) the MPEC model finds ten MPEC-feasible solutions, six of them with zero slack in the validation NLP. Subnetwork operation modes are not fulfilled. The four infeasible instances are correctly identified as MPEC-infeasible, i.e., the MPEC did not converge to a feasible point. The computing time is 29 seconds on average, with median 31, and maximum 37 seconds.

Several typical reasons for unsuccessful runs can be observed. The penalization approach of stage 1 attempts to drive the violations of complementarity constraints to zero. If some violations remain positive in a local minimum, we have an infeasible solution (which may even involve technically impossible states such as „compressing“ control valves). In this case, stage 1 may be repeated with a different regularization scheme that treats complementarity as explicit constraints. Another difficulty arises from numerical inaccuracy: nonzero constraint values that are smaller than the thresholds of the decision heuristics of stage 1 may lead to wrong discrete decisions. For example, very small but non-zero flows into a sub-network behind a valve may result in closing this valve. Finally, the simplified compressor group model may lead to an overly optimistic decision for a compressor group in stage 1. Especially when compressors need to be operated close to the boundary of their operation range, as in the worst-case nominations, this may produce first-stage solutions outside the domain of convergence of stage 2.

5 Comparison and Combination of the Approaches

Sections 4.3, 4.4, 4.5, and 4.6 present an individual analysis of the performance of our four approaches to obtain good discrete decisions and starting points for the NLP validation step. In this section, we compare the different results w.r.t. NLP validation and discuss their combination to yield a reliable solver for NoVa.

5.1 Comparison of the Approaches and their Results

As outlined in Section 3.5, we are validating the solutions of the different solvers using a detailed NLP model. Before passing the solution to the NLP, we check whether all discrete decisions are taken in accordance with the technical restrictions described by the subnetwork operation modes. The latter check is often the reason that a solution of the MPEC approach is rejected. (For the other approaches this test is satisfied by design – and only included to catch implementation errors.) For this reason, we will not compare the MPEC heuristic together with the other approaches in the following.

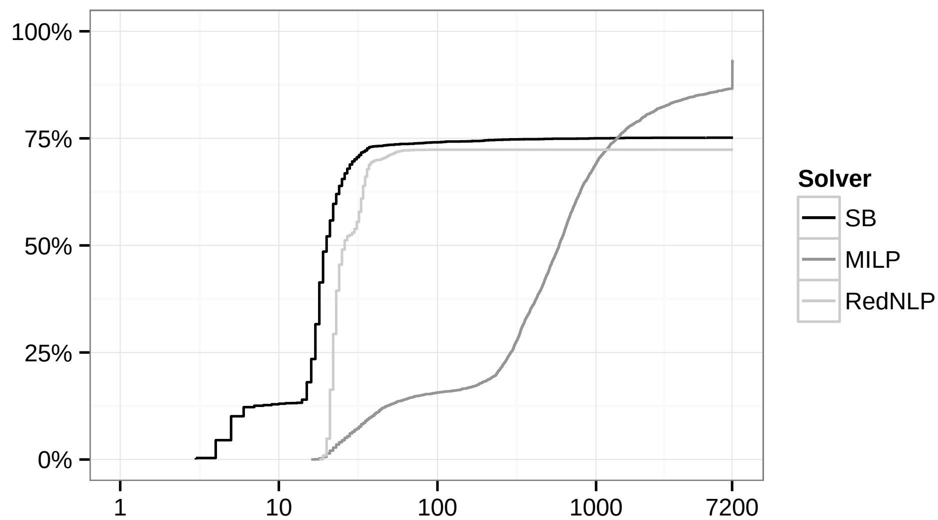


Figure 14: Profile of the run times (s) for test set SN4 including NLP validation; y-axis: percentage of confirmed or infeasible instances (total: 4227)

Table 1: Results of the solvers on test set SN4 (4227 instances)

	slack 0	infeasible	slack > 0	no solution
MILP	3245	694	265	23
SB	2577	600	1036	14
RedNLP	3058	0	1136	33

In line with the description of Section 3.5, we call solutions that yield an NLP slack below the precision of the validation NLP (which allows constraint violations of at most 10^{-5}) *valid* or *confirmed*. We repeat that this might exclude valid solutions, since the validation NLP might only find a local optimum.

On the test set SN4, the overall results of the MILP, SB, and RedNLP approaches are shown in Table 1 and Figure 14. Each solver can report at most one solution to be validated by the detailed NLP. The times now include the time needed for the validation NLP. All instances that are solved by the SB approach within 10 s are infeasible and therefore no validation with the NLP model is carried out. This explains the notable "bump" in the graph of the SB approach in Figure 14. Moreover, when the MILP cannot prove optimality within the time limit, but has found a feasible solution, the best solution found is validated at the very end of the computation. This explains the "jump" of the MILP success at the 7200 s line.

The computations on the test set AB6 are displayed in Table 2 and Figure 15.

When analyzing the results, we note that the SB and the RedNLP approach can both produce solutions for many instances in a rather short time. The MILP approach is by

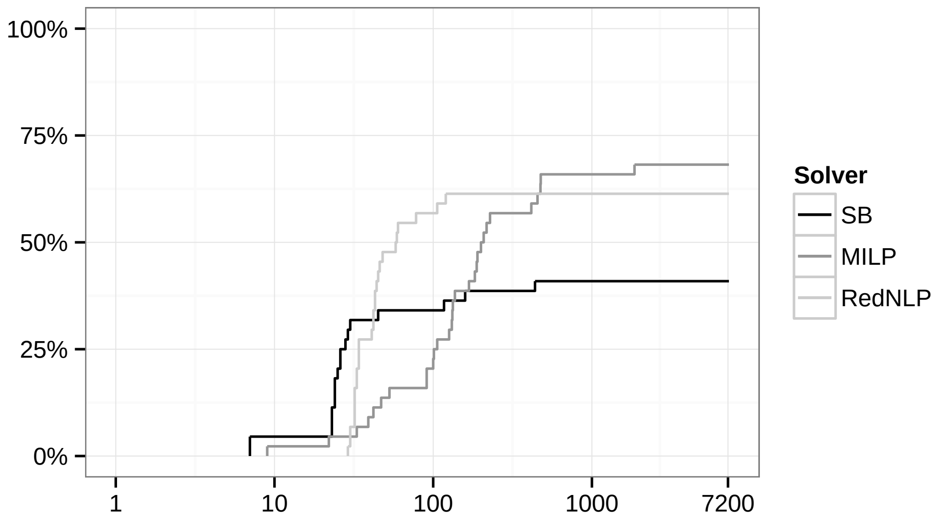


Figure 15: Profile of the run times (s) for test set AB6 including NLP validation; y-axis: percentage of confirmed or infeasible instances (total: 44)

Table 2: Results of the solvers on test set AB6 (44 instances)

	slack 0	infeasible	slack > 0	no solution
MILP	25	5	14	0
SB	14	4	26	0
RedNLP	27	0	12	5

far slower, but has a significantly higher success rate. This is partly due to the parameter setting choice for the MILP approach to yield high accuracy.

The differences between the SB and the RedNLP approach become visible when comparing the numbers in Tables 1 and 2. The SB approach is able to detect infeasibility, whereas the RedNLP approach just reports that no solution has been found.

The differences between the MILP and SB approach with respect to their validation success might be explained as follows. One principal difference in the models is the compressor model. The MILP decides which configurations of the compressor groups are used in the model, while for the SB approach the capability of a compressor group is modeled by the convex hull over all configurations, and a feasible or a "least" infeasible configuration is selected in a postprocessing step. This might include convexification errors and, thus, bad decisions in the surroundings of compressor groups. Apparently, especially on the AB6 instances, it is crucial to choose appropriate configurations for the compressor groups.

The results for the AB6 test set are similar to the ones for the SN4 test set, but the NLP validation results are worse for all approaches. One reason is that the ratio of feasible to infeasible instances is different in the two test sets. It seems to be easier for the SB and the MILP approach to detect infeasibilities than to find primal solutions for challenging instances. This also explains the comparatively better results of the RedNLP approach.

5.2 Combined Solver

We have combined the MILP, SB, and RedNLP approaches presented in Section 3 to form a solver that can be used to reliably solve the NoVa problem for real-world gas network instances as described in the previous sections.

The analysis of the results is complicated by the fact that the models are not direct refinements of each other. All models divert at one or the other point from the model of the validation NLP. For instance, this holds for current implementation of the MILP model of Section 3.1: It could, in principle, be extended to form a proper relaxation of the MINLP corresponding to the validation NLP. This is, however, undesirable in practice: The nondifferentiable functions like sgn and absolute values appearing in the nonlinear models usually have to be smoothed. A MILP model, on the other hand, can model these functions. The smoothing functions, however, may lead to problems in the MILP setting and inexactness. This means, that different models are preferred for the, say, MILP model and the validation NLP.

Consequently, the approaches can contradict each other: one approach might find a feasible solution, while another might report the instance as infeasible. Our approach is, nevertheless, to run all solvers for all models in parallel. As above, each solver can report at most one solution to be validated by the detailed NLP. Recall once again that this approach might exclude valid solutions as we cannot be sure that the validation NLP is not stuck in a local optimum.

For the purpose of analysis, we call such a parallel run *successful*, if the solvers do not contradict each other and either at least one solver finds a solution with zero slack or at least one solver reports infeasible. Otherwise such a run is *unsuccessful*. For the following computational experiments we used the same infrastructure as for the other tests. The parameters of the MILP model of Section 3.1 are calibrated to reliably yield

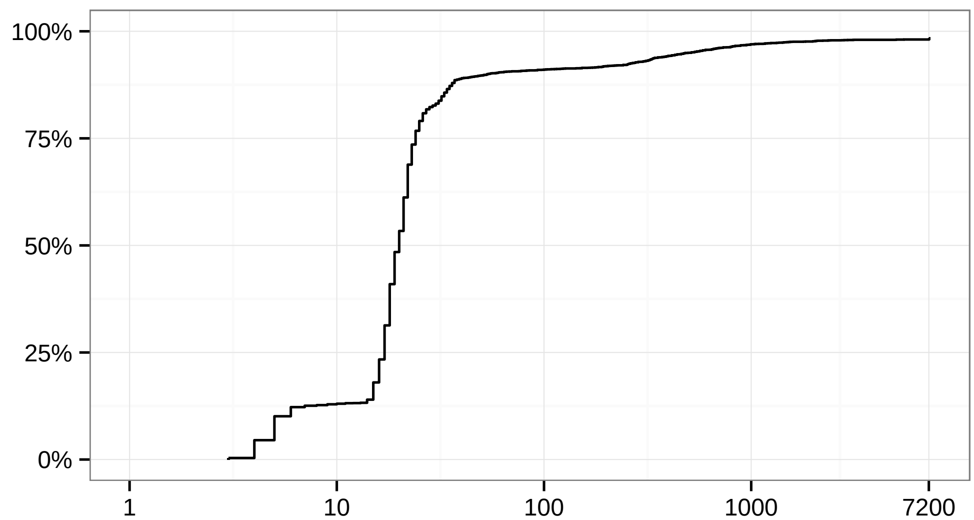


Figure 16: Profile of the run times (s) for the combined approach on test set SN4 including NLP validation; y-axis: percentage of confirmed or infeasible instances (total: 4227)

feasible solutions at the cost of a long running time. We use the other solvers to generate feasible solutions quickly.

The results of the combined solver on the test set SN4 are given in Figure 16. 4157 (more than 98 %) of the 4227 instances in the test set can be solved *successfully*, whereas we are *unsuccessful* on 70 of the instances. Here, 38 of the instances yield a contradictory result, and the remaining 32 failed to yield any definitive result. The times shown in Figure 16 are with respect to the first solver that reports the final result (now including the time needed to validate feasible solutions with the validation NLP).

The results of the combined approach on the test set AB6 are given in Figure 17 (using the same structure as Figure 16). Of the 44 instances in the test set, 38 (more than 86 %) can be solved *successfully*, whereas we are *unsuccessful* on six of the instances. Here, none of the instances yield a contradictory result, and the remaining six fail to yield any definitive result.

6 Summary

Faced with a complex and numerically difficult mixed-integer non-convex nonlinear feasibility problem, we have shown how to utilize the full range of mathematical optimization technology (NLP, MILP, MINLP) in this context.

Our two-stage approach is finally able to successfully solve nearly 98% of the instances with high precision. The paper describes the two key reasons for this success:

- Splitting the problem into two phases: A first phase using approximate models for finding settings for the discrete decisions and then using the results of the first phase

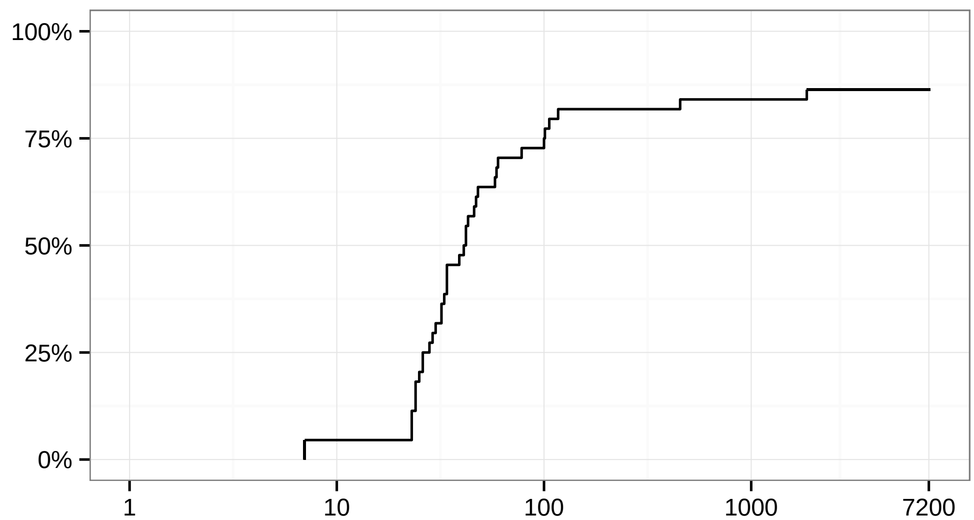


Figure 17: Profile of the run times (s) for the combined approach on test set AB6 including NLP validation; y-axis: percentage of confirmed or infeasible instances (total: 44)

to compute highly precise solutions using an NLP.

- Combination of four quite distinct approaches to solve phase one: While all the approaches in their way are quite successful, none of them for itself is able to come near the combined performance.

Nevertheless, while the networks used in this paper are large in comparison to the state-of-art, they are small compared to the real European pipeline network. We are currently trying to further develop our approach to be able to solve networks which are at least five times bigger.

Another area that leaves room for improvement is the transition from the approximate models to the detailed NLP model. Due to the corresponding model differences, it is not entirely clear what the best objective for the approximate models of the first phase is. In principle, any feasible solution to the approximate models could be tried in the detailed model. This is one of the reasons why we included the MPEC approach. The model used here is the one which is most similar to the detailed NLP model. Consequently, those instances that can be solved typically exhibit no or only very small slacks in the validation. The validation performance of the MPEC approach is currently not comparable to the other approaches, because of the missing SOM constraints. Thus, the MPEC approach is promising, if there is a way to address these issues.

Acknowledgments

We thank Klaus Spreckelsen from Open Grid Europe GmbH for his support and Timo Berthold and Stefan Heinz from the Matheon B20 project for their support on adapting

the SCIP solver.

References

[1] T. Achterberg. SCIP: Solving Constraint Integer Programs. *Mathematical Programming Computation*, 1(1):1–41, 2009.

[2] J. André, F. Bonnans, and L. Cornibert. Optimization of capacity expansion planning for gas transportation networks. *European Journal of Operational Research*, 197(3):1019–1027, 2009.

[3] F. Babonneau, Y. Nesterov, and J.-P. Vial. Design and operations of gas transmission networks. *Operations Research*, 60(1):34–47, 2012.

[4] R. Bagnara, P. M. Hill, and E. Zaffanella. The Parma Polyhedra Library: Toward a Complete Set of Numerical Abstractions for the Analysis and Verification of Hardware and Software Systems. *Science of Computer Programming*, 72(1-2):3–21, 2008.

[5] P. Bales. Hierarchische Modellierung der Eulerschen Flussgleichungen in der Gasdynamik. Master's thesis, Technische Universität Darmstadt, 2005.

[6] P. Belotti, J. Lee, L. Liberti, F. Margot, and A. Wächter. Branching and bounds tightening techniques for non-convex MINLP. *Optimization Methods and Software*, 24(4-5):597–634, 2009.

[7] P. Bonami, L. T. Biegler, A. R. Conn, G. Cornuéjols, I. E. Grossmann, C. D. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, and A. Wächter. An algorithmic framework for convex mixed integer nonlinear programs. *Discrete Optimization*, 5(2):186–204, 2008.

[8] J. Bonnans, G. Spiers, and J.-L. Vie. Global optimization of pipe networks by the interval analysis approach: the belgium network case. Rapport de Recherche RR 7796, INRIA, 2011.

[9] C. Borráz-Sánchez and R. Ríos-Mercado. A Non-Sequential Dynamic Programming Approach for Natural Gas Network Optimization. *WSEAS Transactions on Systems*, 3:1384–1389, 2004.

[10] Bundesministerium der Justiz. Verordnung über den Zugang zu Gasversorgungsnetzen (Gasnetzzugangsverordnung - GasNZV), July 2005.

[11] J. Burgschweiger, B. Gnädig, and M. C. Steinbach. Nonlinear programming techniques for operative planning in large drinking water networks. *The Open Appl. Math. J.*, 3:14–28, 2009.

[12] J. Burgschweiger, B. Gnädig, and M. C. Steinbach. Optimization models for operative planning in drinking water networks. *Optim. Eng.*, 10(1):43–73, 2009.

[13] R. Byrd, J. Nocedal, and R. Waltz. Knitro : An integrated package for nonlinear optimization. In G. Pillo, M. Roma, and P. Pardalos, editors, *Large-Scale Nonlinear Optimization*, volume 83 of *Nonconvex Optimization and Its Applications*, pages 35–59. Springer, 2006.

[14] R. Carter. Compressor station optimization: Computational accuracy and speed. Technical Report PSIG 9605, Pipeline Simulation Interest Group, 1996.

[15] R. Carter, D. Schroeder, and T. Harbick. Some causes and effects of discontinuities in modeling and optimizing gas transmission networks. Technical Report PSIG 9308, Pipeline Simulation Interest Group, 1993.

[16] R. G. Carter. Pipeline optimization: Dynamic programming after 30 years. In *Proceedings of the 30th PSIG Annual Meeting*, 1998.

[17] G. Cerbe. *Grundlagen der Gastechnik*. Technik. Hanser Verlag, Leipzig, Germany, 2008.

[18] A. Chebouba, F. Yalaoui, A. Smati, L. Amodeo, K. Younsi, and A. Tairi. Optimization of Natural Gas Pipeline Transportation Using Ant Colony Optimization. *Computers and Operations Research*, 36:1916–1923, 2009.

[19] C. F. Colebrook. Turbulent flow in pipes with particular reference to the transition region between smooth and rough pipe laws. *Journal of the Institution of Civil Engineers*, 11:133–156, February 1939.

[20] D. de Wolf and Y. Smeers. Optimal dimensioning of pipe networks with application to gas transmission networks. *Operations Research*, 44(4):596–608, July-August 1996.

[21] D. De Wolf and Y. Smeers. The gas transmission problem solved by an extension of the simplex algorithm. *Management Science*, 46(11):1454–1465, November 2000.

[22] P. Domschke, B. Geißler, O. Kolb, J. Lang, A. Martin, and A. Morsi. Combination of Nonlinear and Linear Optimization of Transient Gas Networks. *INFORMS Journal on Computing*, 23:605–617, 2011.

[23] A. S. Drud. CONOPT – a large-scale GRG code. *INFORMS Journal on Computing*, 6(2):207–216, 1994.

[24] K. Ehrhardt and M. C. Steinbach. Nonlinear Optimization in Gas Networks. In H. G. Bock, E. Kostina, H. X. Phu, and R. Ranacher, editors, *Modeling, Simulation and Optimization of Complex Processes*, pages 139–148. Springer, 2005.

[25] European Parliament and Council. Regulation (ec) no. 715/2009: Conditions for access to the natural gas transmission networks. `http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2009:211:0036:0054:en:PDF`, 13 July 2009. Visited 10/2012.

[26] M. Feistauer. *Mathematical Methods in Fluid Dynamics*, volume 67 of *Pitman Monographs and Surveys in Pure and Applied Mathematics Series*. Longman Scientific & Technical, Harlow, 1993.

[27] E. J. Finnemore and J. E. Franzini. *Fluid Mechanics wth Engineering Applications*. McGraw-Hill, 10th edition, 2002.

[28] M. Fukushima and J.-S. Pang. Convergence of a smoothing continuation method for mathematical programming with complementarity constraints. In M. Théra and R. Tichatschke, editors, *Ill-Posed Variational Problems and Regularization Technique, Lecture Notes in Economics and Mathematical Systems*, volume 477, pages 99–110. Springer, 1999.

[29] B. Furey. A Sequential Quadratic Programming-Based Algorithm for Optimization of Gas Networks. *Automatica*, 29(6):1439–1450, 1993.

[30] A. Fügenschuh, B. Geißler, A. Martin, and A. Morsi. The Transport PDE and Mixed-Integer Linear Programming. In C. Barnhart, U. Clausen, U. Lauther, and R. H. Möhring, editors, *Models and Algorithms for Optimization in Logistics*, number 09261 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2009.

[31] A. Fügenschuh, H. Homfeld, H. Schülldorf, and S. Vigerske. Mixed-integer nonlinear problems in transportation applications. In H. R. et al., editor, *Proceedings of the 2nd International Conference on Engineering Optimization (CD-ROM)*, 2010.

[32] General algebraic modeling system (GAMS). `http://www.gams.com/`.

[33] B. Geißler. *Towards Globally Optimal Solutions for MINLPs by Discretization Techniques with Applications in Gas Network Optimization*. PhD thesis, Friedrich-Alexander-Universität Erlangen-Nürnberg, 2011.

[34] B. Geißler, A. Martin, A. Morsi, and L. Schewe. Using piecewise linear functions for solving MINLPs. In J. Lee and S. Leyffer, editors, *Mixed Integer Nonlinear Programming*, volume 154 of *The IMA Volumes in Mathematics and its Applications*, pages 287–314. Springer New York, 2012.

[35] B. Geißler, O. Kolb, J. Lang, G. Leugering, A. Martin, and A. Morsi. Mixed Integer Linear Models for the Optimization of Dynamical Transport Networks. *Mathematical Methods of Operations Research*, 73:339–362, 2011.

[36] P. E. Gill, W. Murray, and M. S. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Journal on Optimization*, 12(4):979–1006, 2002.

[37] B. Gilmour, C. Luongo, and D. Schroeder. Optimization in Natural Gas Transmission Networks: A Tool to Improve Operational Efficiency. Technical report, Stoner Associates Inc., 1989.

[38] Z. Gu, E. Rothberg, and R. Bixby. *Gurobi Optimizer Reference Manual, Version 5.0*. Gurobi Optimization Inc., Houston, USA, 2012.

[39] Y. Hamam and A. Brameller. Hybrid method for the solution of piping networks. *Proc. IEE*, 118(11):1607–1612, 1971.

[40] C. T. Hansen, K. Madsen, and H. B. Nielsen. Optimization of pipe networks. *Mathematical Programming*, 52(1-3):45–58, 1991.

[41] X. M. Hu and D. Ralph. Convergence of a penalty method for mathematical programming with complementarity constraints. *Journal of Optimization Theory and Applications*, 123:365–390, 2004.

[42] IBM Corporation, Armonk, USA. *User's Manual for CPLEX*, 12.4 edition, 2011.

[43] Jerníček. Steady-state optimization of gas transport. In *Proceedings of the 2nd International Workshop SIMONE on Innovative Approaches to Modeling and Optimal Control of Large Scale Pipeline Networks*, 1993.

[44] T. Koch, D. Bargmann, M. Ebbers, A. Fügenschuh, B. Geißler, N. Geißler, R. Gollmer, U. Gotzes, C. Hayn, H. Heitsch, R. Henrion, B. Hiller, J. Humpola, I. Joormann, V. Kühl, T. Lehmann, H. Leövey, A. Martin, R. Mirkov, A. Möller, A. Morsi, D. Oucherif, A. Pelzer, M. E. Pfetsch, L. Schewe, W. Römisch, J. Rövekamp, M. Schmidt, R. Schultz, R. Schwarz, J. Schweiger, K. Spreckelsen, C. Stangl, M. C. Steinbach, A. Steinkamp, I. Wegner-Specht, and B. M. Willert. *From Simulation to Optimization: Evaluating Gas Network Capacities.* Book in preparation, 2012.

[45] T. Koch, H. Leövey, R. Mirkov, W. Römisch, and I. Wegner-Specht. Szenariogenerierung zur Modellierung der stochastischen Ausspeiselasten in einem Gastransportnetz. *VDI-Berichte: Optimierung in der Energiewirtschaft*, 2157:115–125, 2011.

[46] J. Králik. Compressor Stations in SIMONE. In *Proceedings of the 2nd International Workshop SIMONE on Innovative Approaches to Modeling and Optimal Control of Large Scale Pipeline Networks*, 1993.

[47] H. Lall and P. Percell. A dynamic programming based Gas Pipeline Optimizer. In A. Bensoussan and J. Lions, editors, *Analysis and Optimization of Systems*, volume 144 of *Lecture Notes in Control and Information Sciences*, pages 123–132. Springer, 1990.

[48] C. Li, W. Jia, Y. Yang, and X. Wu. Adaptive Genetic Algorithm for Steady-State Operation Optimization in Natural Gas Networks. *Journal of Software*, 6:452–459, 2011.

[49] LIWACOM Informations GmbH and SIMONE Research Group s.r.o. *Gleichungen und Methoden*, 2004. Benutzerhandbuch.

[50] M. V. Lurie. *Modeling of Oil Product and Gas Pipeline Transportation.* Wiley-VCH, 2008.

[51] D. Mahlke, A. Martin, and S. Moritz. A simulated annealing algorithm for transient optimization in gas networks. *Mathematical Methods of Operations Research*, 66:99–116, 2007.

[52] D. Mahlke, A. Martin, and S. Moritz. A mixed integer approach for time-dependent gas network optimization. *Optimization Methods and Software*, 25:625–644, 2010.

[53] J. Mallinson, A. Fincham, S. Bull, J. Rollet, and M. Wong. Methods for optimizing gas transmission networks. *Annals of Operations Research*, 43:443–454, 1993.

[54] H. Markowitz and A. Manne. On the solution of discrete programming problems. *Econometrica*, 25:84–110, 1957.

[55] A. Martin, M. Möller, and S. Moritz. Mixed integer models for the stationary case of gas network optimization. *Mathematical Programming*, 105:563–582, 2006.

[56] M. Möller. *Mixed Integer Models for the Optimisation of Gas Networks in the Stationary Case*. PhD thesis, Technische Universität Darmstadt, 2004.

[57] S. Moritz. *A Mixed Integer Approach for the Transient Case of Gas Network Optimization*. PhD thesis, Technische Universität Darmstadt, 2007.

[58] J. Murdock. *Fundamental Fluid Mechanics for the Practicing Engineer*. Mechanical Engineering. Taylor & Francis, 1993.

[59] A. Osiadacz and S. Swierczewski. Optimal control of gas transportation systems. In *Proceedings of IEEE Conference on Control Applications*, volume 2, pages 795–796, 1994.

[60] A. J. Osiadacz. *Simulation and analysis of gas networks*. Gulf Publishing Company, 1987.

[61] I. Papay. OGIL Musz. Tud. Kozl., 1968.

[62] T. A. Perdicoúlis and L. Fletcher. Decentralised Dynamic Optimisation of Gas. In *31st Annual Meeting of Pipeline Simulation Interest Group*, 1999.

[63] N. Ramchandani. *Optimisation of Gas Networks using Nash Equlibria Derived from Dynamic Non-Cooperative Game Theory*. PhD thesis, Standford University, 1993.

[64] R. Z. Ríos-Mercado and C. Borraz-Sánchez. Optimization Problems in Natural Gas Transmission Systems: A State-of-the-Art Survey. Submitted, January 2012.

[65] R. Z. Ríos-Mercado, S. Wu, L. R. Scott, and E. A. Boyd. A reduction technique for natural gas transmission network optimization problems. *Annals of Operations Research*, 117(1):217–234, 2002.

[66] J. Saleh, editor. *Fluid Flow Handbook*. McGraw-Hill Handbooks. McGraw-Hill, 2002.

[67] M. Schmidt. *A Generic Interior-Point Framework for Nonsmooth and Complementarity Constrained Nonlinear Optimization*. PhD thesis, Gottfried Wilhelm Leibniz Universität Hannover, 2012. In preparation.

[68] M. Schmidt, M. C. Steinbach, and B. M. Willert. High detail stationary optimization models for gas networks — Part 1: Model components. In preparation.

[69] M. Schmidt, M. C. Steinbach, and B. M. Willert. A primal heuristic for nonsmooth mixed integer nonlinear optimization. Submitted.

[70] S. Scholtes. Convergence properties of a regularization scheme for mathematical programs with complementarity constraints. *SIAM Journal on Optimization*, 11(4):918–936, 2001.

[71] SCIP: Solving constraint integer programs., 2012. `http://scip.zib.de/`.

[72] E. Smith and C. Pantelides. A symbolic reformulation/spatial branch-and-bound algorithm for the global optimization of nonconvex MINLPs. *Computers & Chemical Engineering*, 23:457–478, 1999.

[73] M. C. Steinbach. On PDE Solution in Transient Optimization of Gas Networks. *Journal of Computational and Applied Mathematics*, 203(2):345–361, 2007.

[74] O. Strusberg and S. Engell. Optimal Control of Switched Continuous Systems Using Mixed-Integer Programming. In *15th IFAC World Congress of Automatic Control*, 2002.

[75] M. Tawarmalani and N. Sahinidis. *Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming: Theory, Algorithms, Software, and Applications*. Kluwer Academic Publishers, 2002.

[76] M. Tawarmalani and N. Sahinidis. Global optimization of mixed-integer nonlinear programs: A theoretical and computational study. *Mathematical Programming*, 99:563–591, 2004.

[77] M. Tawarmalani and N. Sahinidis. A polyhedral branch-and-cut approach to global optimization. *Mathematical Programming*, 103:225–249, 2005.

[78] S. Vigerske. *Decomposition in Multistage Stochastic Programming and a Constraint Integer Programming Approach to Mixed-Integer Nonlinear Programming*. PhD thesis, Humboldt-Universität zu Berlin, 2012.

[79] Z. Vostrý. Transient Optimization of Gas Transport and Distribution. In *Proceedings of the 2nd International Workshop SIMONE on Innovative Approaches to Modeling and Optimal Control of Large Scale Pipeline Networks*, 1993.

[80] A. Wächter and L. T. Biegler. On the Implementation of a Primal-Dual Interior Point Filter Line Search Algorithm for Large-Scale Nonlinear Programming. *Mathematical Programming*, 106(1):25–57, 2006.

[81] T. Westerlund and R. Pörn. Solving pseudo-convex mixed integer optimization problems by cutting plane techniques. *Optimization and Engineering*, 3:253–280, 2002.

[82] P. J. Wong and R. E. Larson. Optimization of Natural-Gas Pipeline Systems via Dynamic Programming. *IEEE Transactions on Automatic Control*, 15(5):475–481, 1968.

[83] S. Wright, M. Somani, and C. Ditzel. Compressor Station Optimization. Technical report, Pipeline Simulation Interest Group, Denver, Colorado, USA, 1998.

[84] S. Wu, R. Z. Ríos-Mercado, E. A. Boyd, and L. R. Scott. A reduction technique for natural gas transmission network optimization problems. *Mathematical and Computer Modelling*, 31(2):197–220, 2000.

[85] J. Zhang and D. Zhu. A bilevel programming method for pipe network optimization. *SIAM Journal on Optimization*, 6(3):838–857, 1996.

[86] H. Zimmer. Calculating Optimum Pipeline Operations. Technical Report SAND2009-5066C, El Paso Natural Gas Company, 1975.

A Appendix

In this section, we collect detailed results on the 44 instances in the AB6 test set for the MILP, SB, RedNLP, and MPEC approaches described in Section 3, see Tables 3, 4, 5, and 6, respectively. The tables contain data that are specific to the respective approach. In all tables, "Name" refers to the name of the instance, and "Time" shows the time (in seconds) needed for the specific approach, excluding the time for NLP validation. For each approach, "Status" refers to the status of the NLP validation: It displays the value of the NLP validation slack, if the approach produced a solution. If the approach proved infeasibility or found no solution, it displays "inf" or "nosol", respectively. The last column ("Validation Time") displays the time need for NLP validation (in seconds).

Additionally, in Table 3 for the MILP approach, "Nodes" refers to the number of branch-and-bound nodes, "Variables" the total number of variables, "Binary" the number of binary variables, and "Constraints" the number of constraints. Table 4 for the SB approach additionally gives the number of branching on integer variables in "Variable Branchings" and in column "Spatial Branchings" the number of spatial branchings. Table 5 for the RedNLP displays in the column labeled "Transshipment" whether the binary decisions found by the transshipment heuristic lead to a feasible solution. The column "Tested Configs" gives the number of tested preselected configurations. Table 6 for the MPEC approach additionally shows the status after the stage X in "Status X" and the running time of IPOPT in seconds in "Time X".

Table 3: Detailed results of the MILP approach on test set AB6

Name	Time	Status	Nodes	Variables	Binary	Constraints	Validation Time
01-1011	76	0.00	56	14873	5482	23835	9
02-1011	210	0.00	4835	15187	5638	24308	9
04-1011	26	84.01	304	14851	5478	23781	8
03-1011	52	6.80	1904	14805	5455	23712	8
05-1011	123	132.55	4970	14501	5296	23277	8
06-1011	96	8.05	3447	15436	5768	24676	8
07-1011	119	0.00	2331	14843	5463	23778	10
08-1011	437	0.00	42335	14577	5342	23390	10
09-1011	73	0.00	2077	14719	5412	23583	9
09-1011-inf	9	inf	–	–	–	–	–
10-1011	90	0.00	1324	14659	5377	23520	9
01-1112	110	0.00	7674	14865	5477	23820	8
02-1112	189	0.00	15626	15179	5634	24296	9
04-1112	29	86.45	866	14871	5488	23811	8
03-1112	26	71.77	76	14851	5478	23781	8
05-1112	95	55.03	3204	14551	5321	23352	9
06-1112	185	22.88	26767	15432	5766	24670	7
06-1112-inf	39	inf	0	15432	5766	24670	–
07-1112	147	10.13	14837	14835	5459	23766	8
08-1112	1838	0.00	333372	14571	5339	23381	10
09-1112	84	0.00	4411	14865	5485	23802	10
10-1112	168	0.00	17231	14629	5362	23475	8
01-1213	115	0.00	2735	14903	5496	23877	9
02-1213	182	0.00	23920	15175	5632	24290	10
04-1213	26	66.75	402	14871	5488	23811	8
04-1213-inf	22	inf	0	14871	5488	23811	–
04-1213-inf-m	47	inf	5	14908	5505	23887	–
03-1213	24	0.00	0	14801	5453	23706	10
05-1213	33	inf	0	14575	5333	23388	–
06-1213	155	27.86	18846	15334	5717	24523	8
07-1213	115	0.00	6580	14829	5456	23757	9
08-1213	459	0.00	26949	14605	5356	23432	10
09-1213	398	0.00	196466	14863	5484	23799	9
10-1213	173	0.00	13594	14617	5356	23457	9
01-1314	204	0.00	1943	14867	5478	23823	9
02-1314	153	0.00	7797	15171	5630	24284	9
04-1314	24	2.38	0	14873	5489	23814	6
03-1314	26	0.00	0	14801	5453	23706	8
05-1314	77	18.08	1618	14549	5320	23349	8
06-1314	112	28.27	15513	15378	5739	24589	7
07-1314	117	0.00	3584	14829	5456	23757	8
08-1314	458	0.00	9135	14603	5355	23429	10
09-1314	84	0.00	12742	14863	5484	23799	9
10-1314	172	0.00	2067	14615	5355	23454	10

Table 4: Detailed results of the SB approach on test set AB6

Name	Time	Status	Variable Branchings	Spatial Branchings	Nodes	Validation Time
01-1011	10	0.16	0	0	0	10
02-1011	13	34.54	96	1	136	9
04-1011	10	233.00	35	7	53	8
03-1011	10	12.26	38	0	63	9
05-1011	13	656.57	101	0	167	9
06-1011	13	53.45	251	0	439	8
07-1011	12	0.00	92	58	263	10
08-1011	10	526.43	24	0	22	8
09-1011	10	0.09	44	4	59	9
09-1011-inf	7	inf	3	0	1	–
10-1011	9	0.00	17	0	17	10
01-1112	29	0.00	1728	0	3366	10
02-1112	9	0.00	30	0	36	10
04-1112	9	370.67	14	0	16	8
03-1112	9	190.47	49	0	80	9
05-1112	100	0.00	4449	4181	17194	10
06-1112	10	0.16	122	0	219	8
06-1112-inf	438	inf	37548	0	74648	–
07-1112	14	0.00	202	0	341	9
08-1112	12	187.67	161	0	243	10
09-1112	8	10.04	22	0	22	8
10-1112	8	0.00	22	1	21	9
01-1213	10	34.58	15	0	17	10
02-1213	11	0.00	63	0	109	10
04-1213	9	2.21	46	0	71	9
04-1213-inf	7	inf	0	0	0	–
04-1213-inf-m	159	inf	11653	0	23042	–
03-1213	10	8.89	75	0	133	12
05-1213	48	419.13	2255	0	4325	9
06-1213	9	4.16	25	0	23	9
07-1213	9	36.65	18	0	18	14
08-1213	9	0.00	11	0	13	9
09-1213	7	13.17	14	0	14	8
10-1213	9	0.00	22	0	22	10
01-1314	9	36.74	12	0	13	9
02-1314	10	0.00	70	0	123	9
04-1314	9	9.32	49	0	78	8
03-1314	10	10.54	15	0	16	14
05-1314	58	533.70	3464	0	6790	9
06-1314	12	0.11	142	1	242	9
07-1314	12	0.00	87	0	134	10
08-1314	9	0.00	13	0	13	10
09-1314	10	0.02	24	9	37	9
10-1314	10	0.00	25	0	23	10

Table 5: Detailed results of the RedNLP approach on test set AB6

Name	Time	Status	Transshipment	Tested Configs	Validation Time
01-1011	17	0.00	Yes	0	9
02-1011	15	0.00	Yes	0	9
04-1011	27	0.00	No	4	7
03-1011	24	0.00	No	4	8
05-1011	29	166.59	No	2	9
06-1011	76	nosol	–	33	–
07-1011	17	0.00	Yes	0	8
08-1011	12	436.10	Yes	0	44
09-1011	59	0.00	No	26	9
09-1011-inf	98	nosol	–	33	–
10-1011	17	0.00	Yes	0	9
01-1112	16	0.00	Yes	0	8
02-1112	14	0.00	Yes	0	8
04-1112	41	0.00	No	18	9
03-1112	27	0.00	No	4	9
05-1112	21	27.07	Yes	0	7
06-1112	18	153.81	No	1	8
06-1112-inf	78	nosol	–	33	–
07-1112	17	0.00	Yes	0	9
08-1112	13	560.38	Yes	0	7
09-1112	101	0.00	No	26	9
10-1112	16	0.00	Yes	0	8
01-1213	17	0.00	Yes	0	10
02-1213	14	0.00	Yes	0	8
04-1213	41	0.00	No	18	9
04-1213-inf	80	nosol	–	33	–
04-1213-inf-m	82	nosol	–	33	–
03-1213	26	0.00	No	4	9
05-1213	23	35.78	Yes	0	8
06-1213	18	0.11	No	1	8
07-1213	19	0.00	Yes	0	9
08-1213	13	279.50	Yes	0	8
09-1213	86	0.00	No	26	9
10-1213	14	0.00	Yes	0	9
01-1314	17	0.00	Yes	0	8
02-1314	15	0.00	Yes	0	9
04-1314	33	0.00	No	4	8
03-1314	26	0.00	No	4	9
05-1314	27	225.91	No	2	8
06-1314	17	123.70	No	1	8
07-1314	17	0.00	Yes	0	9
08-1314	12	292.22	Yes	0	9
09-1314	28	0.03	No	4	8
10-1314	14	0.00	Yes	0	8

Table 6: Detailed results of the MPEC approach on test set AB6 (instances with violated SOMs in *italics*)

Name	Time	Status	Status 1	Time 1	Status 2	Time 2	Validation Time
01-1011	33	nosol	optimal	3	infeasible	2	–
02-1011	35	*0.00*	optimal	4	optimal	1	9
04-1011	36	nosol	optimal	4	infeasible	2	–
03-1011	29	nosol	optimal	2	infeasible	2	–
05-1011	36	nosol	optimal	5	infeasible	1	–
06-1011	37	nosol	optimal	4	infeasible	7	–
07-1011	35	*55.28*	optimal	3	optimal	2	11
08-1011	35	nosol	optimal	5	infeasible	1	–
09-1011	28	nosol	optimal	2	optimal	2	–
09-1011-inf	21	nosol	infeasible	3	–	–	–
10-1011	36	nosol	optimal	4	infeasible	1	–
01-1112	32	nosol	optimal	3	optimal	1	–
02-1112	34	nosol	optimal	4	infeasible	3	–
04-1112	29	nosol	optimal	3	optimal	1	–
03-1112	32	nosol	optimal	3	optimal	1	–
05-1112	31	nosol	optimal	2	infeasible	1	–
06-1112	36	nosol	optimal	5	infeasible	3	–
06-1112-inf	32	nosol	optimal	5	infeasible	1	–
07-1112	33	nosol	optimal	2	optimal	2	–
08-1112	31	nosol	optimal	4	infeasible	1	–
09-1112	32	nosol	optimal	4	infeasible	2	–
10-1112	34	*1.62*	optimal	5	optimal	1	8
01-1213	32	nosol	optimal	1	infeasible	2	–
02-1213	31	*0.00*	optimal	3	optimal	1	10
04-1213	32	nosol	optimal	3	infeasible	2	–
04-1213-inf	29	nosol	optimal	2	infeasible	2	–
04-1213-inf-m	30	nosol	optimal	2	infeasible	1	–
03-1213	32	nosol	optimal	4	optimal	1	–
05-1213	34	*978.74*	optimal	4	optimal	2	9
06-1213	32	nosol	optimal	3	infeasible	2	–
07-1213	24	*0.00*	optimal	4	optimal	1	10
08-1213	22	*1618.63*	optimal	3	optimal	1	9
09-1213	20	nosol	optimal	4	optimal	0	–
10-1213	22	nosol	optimal	2	infeasible	1	–
01-1314	22	*0.00*	optimal	2	optimal	1	8
02-1314	22	nosol	optimal	2	infeasible	1	–
04-1314	19	nosol	optimal	3	optimal	1	–
03-1314	22	nosol	optimal	4	optimal	1	–
05-1314	24	nosol	optimal	3	infeasible	2	–
06-1314	26	nosol	optimal	7	infeasible	2	–
07-1314	24	*0.00*	optimal	4	optimal	1	9
08-1314	25	nosol	optimal	5	infeasible	2	–
09-1314	23	nosol	optimal	6	infeasible	1	–
10-1314	25	*0.00*	optimal	5	optimal	1	8