

Open Journal of Mathematical Optimization

Felix Hennings, Kai Hoppmann-Baum & Janina Zittel

Optimizing transient gas network control for challenging real-world instances using MIP-based heuristics

Volume 5 (2024), article no. 1 (34 pages)

<https://doi.org/10.5802/ojmo.29>

Article submitted on May 24, 2022, revised on January 17, 2023,
accepted on February 14, 2024.

© The author(s), 2024.



This article is licensed under the

CREATIVE COMMONS ATTRIBUTION 4.0 INTERNATIONAL LICENSE.


<http://creativecommons.org/licenses/by/4.0/>




Optimizing transient gas network control for challenging real-world instances using MIP-based heuristics

Felix Hennings 

Technische Universität Berlin
Chair of Software and Algorithms for Discrete Optimization
Straße des 17. Juni 135, 10623 Berlin, Germany
<https://ror.org/03v4gjf40>
hennings@zib.de

Kai Hoppmann-Baum 

Zuse Institute Berlin
Applied Algorithmic Intelligence Methods Department
Takustraße 7, 14195 Berlin, Germany
<https://ror.org/02eva5865>
hoppmann-baum@zib.de

Janina Zittel 

Zuse Institute Berlin
Applied Algorithmic Intelligence Methods Department
Takustraße 7, 14195 Berlin, Germany
<https://ror.org/02eva5865>
zittel@zib.de

Abstract

Optimizing the transient control of gas networks is a highly challenging task. The corresponding model incorporates the combinatorial complexity of determining the settings for the many active elements as well as the non-linear and non-convex nature of the physical and technical principles of gas transport. In this paper, we present the latest improvements of our ongoing work to tackle this problem for real-world, large-scale problem instances: By adjusting our mixed-integer non-linear programming model regarding the gas compression capabilities in the network, we reflect the technical limits of the underlying units more accurately while maintaining a similar overall model size. In addition, we introduce a new algorithmic approach that is based on splitting the complexity of the problem by first finding assignments for discrete variables and then determining the continuous variables as locally optimal solution of the corresponding non-linear program. For the first task, we design multiple different heuristics based on concepts for general time-expanded optimization problems that find solutions by solving a sequence of sub-problems defined on reduced time horizons. To demonstrate the competitiveness of our approach, we test our algorithm on particularly challenging historical demand scenarios. The results show that high-quality solutions are obtained reliably within short run times, making the algorithm well-suited to be applied at the core of time-critical industrial applications.

Digital Object Identifier 10.5802/ojmo.29

Keywords Transient Gas Network Optimization, Sequential Mixed-Integer Programming, Rolling Horizon Heuristic, Aggregated Horizon Heuristic, Real-World Historical Instances, Industry-Ready.

Acknowledgments The work for this article has been conducted in the Research Campus MODAL, funded by the German Federal Ministry of Education and Research (BMBF) (fund numbers 05M14ZAM & 05M20ZBM).

1 Introduction

The natural gas supply of Europe and especially Germany is subject to a disruptive transformation. By the cut of the Russian gas supply and the recently solidified plans to build LNG terminals in Germany as published by the government [11], new flow patterns are likely to emerge in the corresponding transport networks. Moreover, the usage of fossil-based natural gas is planned to be terminated in favor of more sustainable alternatives as part of the energy transformation towards massive reductions in CO₂ emissions. In particular, the natural gas networks may be used to store synthesized methane generated via power-to-gas from excess renewable energy or could (partially) be repurposed for hydrogen transport, see, for example, the scenario framework for the German *Gas*



© Felix Hennings & Kai Hoppmann-Baum & Janina Zittel;
licensed under Creative Commons License Attribution 4.0 International

Network Development Plan as published by the German gas network operators [12]. Especially when applying power-to-gas strategies, the intraday supply patterns are likely to become more volatile, complicating the control of the networks even further.

Mathematical optimization can assist the network operators in preparing for these new challenges and ensuring a secure and efficient network operation. However, solving the corresponding optimization problems is hard. On the one hand, the physics of gas transport in pipelines describing the interplay of gas pressure and flow is captured by equations having a non-convex feasible set. On the other hand, the network contains many active elements to control the gas flow, like valves to dynamically change the network topology and compressors to increase the gas pressure. Apart from the non-convex description of the maximum power available to compressors, these control options also introduce an immense combinatorial complexity to the problem. Since the problem is both practically relevant and theoretically challenging, it has been intensively studied in the past. In the following, we will give an overview split into two categories, where we focus on approaches featuring discrete decisions.

First, there is the *stationary* variant of the problem, which searches for an equilibrium state of the network with time-constant supply, demand, and control settings and can, for example, be used for planning purposes. For a profound introduction to stationary gas network optimization, we recommend the book of [27] or the paper of [37] for a summary of the corresponding findings. They model the problem as a mixed-integer non-linear program (MINLP) and deal with the two sources of complexity separately by first finding solutions for the discrete decision variables and afterward completing these to highly precise solutions by solving a corresponding non-linear programming (NLP) model. A more recent study on the stationary problem is presented in [16]. Here, the authors mainly focus on solving an NLP-model variant of the gas network optimization problem with a two-step sequential linear programming (SLP) approach. However, by replacing the first step with sequential mixed-integer programming (SMIP), they can convert their algorithm into a heuristic for MINLP problems.

Second, the *transient* variant of the problem aims at finding a short-term control for the network over discrete future time steps. An early publication on the topic was the thesis of [31], where the problem was formulated as a pure mixed-integer program (MIP) by approximating the non-linear pipe equations and compression power constraints using piece-wise linear functions. To aid the corresponding solver, the author proposed a special branching scheme for the piece-wise linear functions and used a simulated annealing heuristic, which was also described in [30]. Shortly after, a similar problem formulation was considered by [8], which again used piece-wise linear functions to approximate the non-linear parts of the model. They iteratively improved the linearization by first solving the corresponding MIP model, then using the binary solution values to form an NLP model featuring the original non-linear equations, and afterward refining the linearization by adding mesh-points corresponding to the found NLP model solutions. In a more recent study, [19] directly solve an MINLP formulation of the problem. Although the problem is non-convex, they use a convex solver and apply a customized branching node selection rule for the problem. In [17], the authors apply the approach of finding an instantaneous control for a transient gas network optimization problem with discrete decisions, i.e., a control that only looks one time step ahead when making a decision. They iteratively solve a MIP model for each of these single time steps, which allows for very short run times despite using a rather fine time and space granularity in their experiments.

Another contribution was made by [6]. By introducing a new discretization for the differential equations describing the gas flow in pipes, they were able to reformulate the transient problem as a stationary one on a time-expanded graph and use a specialized algorithm designed for the stationary case to solve the problem. It is based on iteratively solving and refining MIP relaxations of the problem, which they again obtained by applying a piece-wise linearization.

Finally, we mention the work of [25], who introduced the problem discussed in this paper. The authors describe a decision support system for network operators to ensure a secure and efficient network operation. The system is split into two separate stages: First, an MINLP problem formulation covering the complete network is solved, which approximates the capabilities of the technical elements located at large pipeline intersections, the *network stations*. In particular, each of these stations is replaced with a simplified graph model using artificial arcs. Non-technical control measures available to the network operators are realized as slack variables to adjust the future supplies and demands in terms of pressure and inflow. Their usage as well as changes in the technical control are penalized in the objective functions of a tri-level optimization model. To derive a feasible solution, a series of linearized MIP models are solved, which use two types of primal heuristics to determine good initial solutions. Afterward, a linear programming (LP) model is solved to smooth the solution, followed by an SLP approach that transforms the solution into one respecting the original non-linear constraints of the MINLP

problem. In a computational study on subsequent real-world instances spanning multiple days, the algorithm’s run time and the differences in the solutions are evaluated. In the second stage of the decision support system, the actual control measures recommended to the network operators are determined by solving highly detailed models for each network station individually, as described in [22]. Note that since the overall system runs in regular intervals to provide up-to-date control recommendations, it must reliably satisfy strict time limit restrictions.

In this paper, we present several improvements to the simplified network-wide model of the first stage of the decision support system described in [25]: We suggest an alternative model for the artificial network station arcs representing the gas compressors, which is closer to the original compression capabilities of the highly detailed network station model used in the second stage without increasing the model’s size on real-world instances. Furthermore, we formulate a novel algorithm for tackling the updated MINLP problem of the first stage, which finds solutions of high quality in short amounts of time and thereby outperforms an algorithm designed according to [25]. In particular, we first determine sets of values for all binary variables of the model, which we call *binary assignments*, and afterward complete these to full solutions of the MINLP by solving the remaining NLP model. The binary assignments are obtained by combining an SMIP routine with two heuristic concepts for time-expanded optimization problems: the rolling horizon and the aggregate horizon approach. We additionally apply a dynamic branch-and-bound node limit to the single model solves to further reduce maximum run times. Finally, we thoroughly analyze our proposed algorithm’s solution quality using valid lower bounds and evaluate its run time. As the test set, we identify especially hard-to-solve real-world instances of the problem that feature volatile demand scenarios and are challenging for the dispatchers as well. Thereby, we verify that our algorithm is suitable for time-critical industry applications, even for difficult transport situations.

The approach we propose focuses on finding good primal solutions fast and does not provide a valid lower bound on the value of the problem’s optimal solution. This enables us to tackle larger and more challenging problem instances than those found in the literature while respecting strict time limits. We follow the idea of dealing with the combinatorial and non-linear aspects of the problem separately and also use sequential mixed-integer programming for finding good binary assignments. Since we do not aim at finding lower bounds, we do not need to use an iteratively refined piece-wise linear formulation but only an updatable linearization, similar to what [16] propose for the first level of their algorithm. As a consequence, the general size of the MIP problems that we are going to solve is smaller and does not increase during the solving process.

The heuristic concepts we use for time-expanded problems are well-known in the literature. The rolling horizon approach was, for example, used for the optimization of product manufacturing in [41] or distributed power generation in [7]. Regarding gas network optimization, we mention the already discussed work of [17], whose instantaneous control approach is equivalent to a one-step rolling horizon, as well as [25], who uses a rolling horizon approach as one of their primal heuristics. A theoretical analysis of rolling horizon approaches, as well as a more extensive list of applications, is given in [15]. Similarly, heuristics using an aggregation of a time horizon were, for example, described in the scheduling of iron ore production in [32] or in the extension of freight transport networks in [4].

The rest of the paper is structured as follows: Section 2 provides the complete MINLP problem formulation. In Section 3, we present our heuristic algorithmic approach for tackling this model. The computational results are the content of Section 4, where we describe the creation of the instance sets, compare a base version of our algorithm with a global MINLP solver in terms of solution quality on smaller instances, and then compare the actually proposed algorithm with the base algorithm version on a more challenging instance set. We close with concluding remarks in Section 5.

2 Model formulation

Our model is based on the gas flow model used for the first stage of the decision support system presented in [25]. Hence, we refer to it as the *base model* for the remainder of the paper and highlight the differences between both models in Section 2.12 below. However, note this paper provides all model details and is self-sufficient in this sense.

As in [25], we assume that the gas network contains a set \mathcal{I} of *network stations*. Those are sub-networks that include the majority of all actively controllable network elements, among them all the network’s compressor stations, and are mainly located at large pipeline intersections. For the model, the inner network topology of each station is replaced by a set of artificial arcs representing an approximation of the station’s gas routing,

compression, and regulation capabilities. This artificial model was created manually by industry experts who work for the network operator and have an excellent knowledge of the individual stations and their properties.

When formulating our model below, we often use *indicator constraints* as, for example, introduced in [5]. A list of all the variables used in our model, together with their domains, meanings, and units, is given in Table 6 in the Appendix.

2.1 Gas flow in networks

We model a gas network as a directed graph $G = (\mathcal{V}, \mathcal{A})$ with \mathcal{V} being the set of nodes and $\mathcal{A} = \mathcal{A}^{\text{pi}} \dot{\cup} \mathcal{A}^{\text{va}} \dot{\cup} \mathcal{A}^{\text{rg}} \dot{\cup} \mathcal{A}^{\text{ar}}$ being the set of arcs representing the single element sets present in the network: pipes \mathcal{A}^{pi} , valves \mathcal{A}^{va} , regulators \mathcal{A}^{rg} , and artificial network station arcs \mathcal{A}^{ar} . The latter is composed of the different types of artificial arcs found in the network stations: shortcuts $\mathcal{A}^{\text{arSc}}$, regulating arcs $\mathcal{A}^{\text{arRg}}$, compressor arcs $\mathcal{A}^{\text{arCp}}$, and bi-regulating arcs $\mathcal{A}^{\text{arBiRg}}$. For station $i \in \mathcal{I}$, we denote by $\mathcal{A}_i^{\text{ar}} = \mathcal{A}_i^{\text{arSc}} \dot{\cup} \mathcal{A}_i^{\text{arRg}} \dot{\cup} \mathcal{A}_i^{\text{arBiRg}} \dot{\cup} \mathcal{A}_i^{\text{arCp}}$ its set of artificial arcs and by $\mathcal{V}_i^{\text{fn}} \subseteq \mathcal{V}$ the nodes forming the station's boundary, which we call *fence nodes*. Formally, the fence nodes are those nodes having incident arcs from within the station as well as from outside of the station. However, a node can only be the fence node of a single station. Finally, we assume for each station $i \in \mathcal{I}$ that the graph induced by the station's arcs $\mathcal{A}_i^{\text{ar}}$ is connected.

We consider a discrete time horizon $\mathcal{T}_0 := \{0, \dots, t^{\text{max}}\}$, where $\mathcal{T} := \mathcal{T}_0 \setminus \{0\}$ denotes the set of all future time points. We denote by $\tau(t)$ the time difference in seconds from timestep $t \in \mathcal{T}_0$ to the initial state time point 0.

The gas flow on each arc $a = (\ell, r)$ from node ℓ to node r at some time $t \in \mathcal{T}_0$ is characterized by the pressure p and the mass flow q at each end of the arc, i.e., the pressure at the left node $p_{\ell,t}$, the pressure at the right node $p_{r,t}$, the incoming mass flow $q_{\ell,a,t}$, and the outgoing mass flow $q_{r,a,t}$. Here, negative flow values represent flow against the arc's orientation, while the pressure values are always positive. Since we assume that only pipes have a non-zero volume, the inflow and outflow of all non-pipe arcs is identical. Thus, we define the mass flow of an arc $a \in \mathcal{A} \setminus \mathcal{A}^{\text{pi}}$ as $q_{a,t} := q_{\ell,a,t} = q_{r,a,t}$.

At each node holds the conservation of mass or the *flow balance equation*, a Kirchhoff-type law which demands that the flow into and out of a node $v \in \mathcal{V}$ must be balanced for all time points $t \in \mathcal{T}$, i.e.,

$$\sum_{a \in \mathcal{A}: a=(v,r)} q_{v,a,t} - \sum_{a \in \mathcal{A}: a=(\ell,v)} q_{v,a,t} = d_{v,t}. \quad (1)$$

Note that the inflow $d_{v,t}$ denotes gas injection into node v at time t from outside of the network for positive values and gas withdrawal for negative values. Furthermore, the inflow value at time t represents the average gas inflow rate into the node in the continuous time interval between $t - 1$ and t . We partition the nodes into three sets: the set of *entries* \mathcal{V}^+ with $d_{v,t} \geq 0$ for $v \in \mathcal{V}^+$, the set of *exits* \mathcal{V}^- with $d_{v,t} \leq 0$ for $v \in \mathcal{V}^-$, and the set of inner nodes \mathcal{V}^0 with $d_{v,t} = 0$ for $v \in \mathcal{V}^0$ for all $t \in \mathcal{T}$. We call the union of all entries and exits the *boundary nodes* \mathcal{V}^{b} of the network.

For each pressure and flow variable, there exist technical lower and upper bounds for each $t \in \mathcal{T}$, which are denoted by bars below and above the corresponding variable. Furthermore, we are given the initial state of the network, which specifies for $t = 0$ the pressure $p_{v,0}$ for nodes $v \in \mathcal{V}$ and the flow values $q_{\ell,a,0}$ and $q_{r,a,0}$ for arcs $a = (\ell, r) \in \mathcal{A}$. Additionally, the initial states of the active network elements outside of the network stations, i.e., valves and regulators, are known.

2.2 Pipes

For pipes, we use the *friction-dominated* model given as (ISO3) in [9]. It is based on the *Euler equations* modeling the one-dimensional gas flow in straight cylindrical pipes and can be derived by assuming a constant temperature (isothermal conditions), gas composition, and compressibility factor, as well as gas velocities much smaller than the speed of sound and a negligible inertia term, which both are usually given in real-world conditions, see [35, 9] and [21]. Note that while we assume a constant gas temperature over time, we may consider different temperature levels for different parts of the network to represent changing soil structure or varying depth levels of the pipelines. Hence, there may exist pipelines that have different constant temperature levels but are incident to the same node. This would lead to a discontinuity of temperature at this node: a temperature shock. However, since there are usually only a few different constant temperature levels and the corresponding network areas are often linked by active elements, the vast majority of connected pipes have the same mutual constant temperature level. The usage of the *implicit box* scheme of [8, 28] as a discretization yields the model for a pipe $a = (\ell, r) \in \mathcal{A}^{\text{pi}}$ and two

adjacent time points $t_1 \in \mathcal{T}$ and $t_0 := t_1 - 1$ given as

$$p_{\ell,t_1} + p_{r,t_1} - p_{\ell,t_0} - p_{r,t_0} + \frac{2R_s T_a z_a (\tau(t_1) - \tau(t_0))}{L_a A_a} (q_{r,a,t_1} - q_{\ell,a,t_1}) = 0, \quad (2a)$$

$$p_{r,t_1} - p_{\ell,t_1} + \underbrace{\frac{\lambda_a R_s T_a z_a L_a}{4A_a^2 D_a} \left(\frac{|q_{\ell,a,t_1}| q_{\ell,a,t_1}}{p_{\ell,t_1}} + \frac{|q_{r,a,t_1}| q_{r,a,t_1}}{p_{r,t_1}} \right)}_{\text{friction term}} + \frac{g s_a L_a}{2R_s T_a z_a} (p_{\ell,t_1} + p_{r,t_1}) = 0. \quad (2b)$$

The first equation (2a) is called the *continuity* equation, while the second equation (2b) is known as the *momentum* equation and contains the non-linear *friction term*. In the equations, the specific gas constant is given by R_s , the gravitational acceleration by g , the gas temperature of the pipe by T_a , its compressibility factor by z_a , its length by L_a , its diameter by D_a , its area by A_a defined as $A_a = D_a^2 \pi / 4$, its Darcy friction coefficient by λ_a , and finally its slope by s_a defined as $s_a = (h_r - h_\ell) / L_a$ using the height or altitude h of the pipes' endnodes. For the gas temperature, we are given constant values T_v for each endnode $v \in \{\ell, r\}$ of the pipe and determine the pipe's constant value as average of both. In a similar fashion, we determine the pipes' compressibility factor as the average of those of the two endnodes by using the formula of Pápay [36, 39] based on the initial pressure and the constant temperature of each node. Finally, the Darcy friction factor λ_a is determined using the formula of Nikuradse [33], which is an explicit approximation depending only on the pipe's diameter and its integral roughness k_a .

2.3 Valves and regulators

Valves are active elements that are able to dynamically connect or disconnect two network nodes. Their state is represented by a binary mode variable $m_{a,t}^{\text{op}}$, which specifies if the valve a is open ($m_{a,t}^{\text{op}} = 1$) at time t , connecting both endnodes and forcing their pressures variables to attain identical values, or closed ($m_{a,t}^{\text{op}} = 0$), disconnecting the nodes, decoupling the respective pressure values, and prohibiting gas flow exchange. Thus, the corresponding model for a valve $a = (\ell, r) \in \mathcal{A}^{\text{va}}$ reads

$$m_{a,t}^{\text{op}} = 1 \implies p_{\ell,t} = p_{r,t} \quad \forall t \in \mathcal{T}, \quad (3a)$$

$$m_{a,t}^{\text{op}} = 0 \implies q_{a,t} = 0 \quad \forall t \in \mathcal{T}, \quad (3b)$$

$$m_{a,t}^{\text{op}} \in \{0, 1\} \quad \forall t \in \mathcal{T}_0.$$

Regulators are active elements that can be seen as an extension of valves. In addition to changing the network's connectivity, they can reduce the pressure of the gas in the direction of the flow. However, this is only possible along the arc's orientation. We use the regulator model of [22] here, in which the state of a regulator a at time t is modeled by a set of three different binary mode variables: The two valve state variables of an open regulator $m_{a,t}^{\text{op}}$ and a closed regulator $m_{a,t}^{\text{cl}}$ as well as the active mode $m_{a,t}^{\text{ac}}$, which allows for pressure reduction. The model for all $a = (\ell, r) \in \mathcal{A}^{\text{rg}}$ can be stated as

$$1 = m_{a,t}^{\text{ac}} + m_{a,t}^{\text{op}} + m_{a,t}^{\text{cl}} \quad \forall t \in \mathcal{T}, \quad (4a)$$

$$m_{a,t}^{\text{op}} = 1 \implies p_{\ell,t} \leq p_{r,t} \quad \forall t \in \mathcal{T}, \quad (4b)$$

$$m_{a,t}^{\text{ac}} + m_{a,t}^{\text{op}} = 1 \implies p_{\ell,t} \geq p_{r,t} \quad \forall t \in \mathcal{T}, \quad (4c)$$

$$m_{a,t}^{\text{cl}} = 1 \implies q_{a,t} \leq 0 \quad \forall t \in \mathcal{T}, \quad (4d)$$

$$q_{a,t} \geq 0 \quad \forall t \in \mathcal{T}, \quad (4e)$$

$$m_{a,t}^{\text{op}}, m_{a,t}^{\text{cl}}, m_{a,t}^{\text{ac}} \in \{0, 1\} \quad \forall t \in \mathcal{T}_0.$$

Note that we assume an internal check valve in the element, which prevents flow against the orientation even in the case of an open regulator. For a more detailed description of the internal regulator behavior, we refer to [23].

2.4 Artificial non-compressing station arcs

As stated above, the original network topology inside the stations is replaced by artificial arcs of different types, which represent an approximation of the station's gas transport capabilities. Each arc can be active or inactive, which we model by a binary variable

$$x_{a,t}^{\text{arc}} \in \{0, 1\} \quad \forall a \in \mathcal{A}^{\text{ar}} \quad \forall t \in \mathcal{T}_0, \quad (5)$$

where $x_{a,t}^{\text{arc}} = 1$ represents the active state. An inactive arc behaves like a closed valve, i.e., the flow is zero and the endnodes' pressures are decoupled, while the active behavior is specific to each artificial arc type, which we discuss in the following. As first type of artificial arcs there are shortcuts, which connect its endnodes if active. Hence, their model is equivalent to that of a valve and can be stated for a shortcut $a = (\ell, r) \in \mathcal{A}^{\text{arSc}}$ and time $t \in \mathcal{T}$ as:

$$x_{a,t}^{\text{arc}} = 1 \implies p_{\ell,t} = p_{r,t}, \quad (6a)$$

$$x_{a,t}^{\text{arc}} = 0 \implies q_{a,t} = 0. \quad (6b)$$

Artificial regulating arcs capture the general behavior of regulators. Hence, they only allow flow in the direction of their orientation and offer the possibility to reduce the pressure along that orientation if they are active. The corresponding model for a regulating arc $a = (\ell, r) \in \mathcal{A}^{\text{arRg}}$ and time $t \in \mathcal{T}$ can be stated as

$$x_{a,t}^{\text{arc}} = 1 \implies p_{\ell,t} \geq p_{r,t}, \quad (7a)$$

$$x_{a,t}^{\text{arc}} = 0 \implies q_{a,t} \leq 0, \quad (7b)$$

$$q_{a,t} \geq 0. \quad (7c)$$

To represent the behavior of a pair of anti-parallel regulators in the original network topology, we use a bi-regulating arc. We can model its behavior by choosing an orientation of the element in active mode and applying the corresponding constraints of a regulating arc for each direction separately. The model for a bi-regulating arc $a = (\ell, r) \in \mathcal{A}^{\text{arBiRg}}$ and time $t \in \mathcal{T}$ is given as

$$x_{a,t}^{\text{arc}} = x_{a,t}^{\text{fwd}} + x_{a,t}^{\text{bwd}}, \quad (8a)$$

$$q_{a,t} = q_{a,t}^{\text{fwd}} - q_{a,t}^{\text{bwd}}, \quad (8b)$$

$$x_{a,t}^{\text{fwd}} = 1 \implies p_{\ell,t} \geq p_{r,t}, \quad (8c)$$

$$x_{a,t}^{\text{fwd}} = 0 \implies q_{a,t}^{\text{fwd}} \leq 0, \quad (8d)$$

$$q_{a,t}^{\text{fwd}} \geq 0, \quad (8e)$$

$$x_{a,t}^{\text{bwd}} = 1 \implies p_{r,t} \geq p_{\ell,t}, \quad (8f)$$

$$x_{a,t}^{\text{bwd}} = 0 \implies q_{a,t}^{\text{bwd}} \leq 0, \quad (8g)$$

$$q_{a,t}^{\text{bwd}} \geq 0, \quad (8h)$$

$$x_{a,t}^{\text{fwd}}, x_{a,t}^{\text{bwd}} \in \{0, 1\}.$$

Here, $x_{a,t}^{\text{fwd}}$ represents the choice for the *forward direction* for arc a and time t , in which flow and regulation happen according to the arc's orientation, while $x_{a,t}^{\text{bwd}}$ represents the choice of the *backward direction*, in which the direction of the flow is reversed. Analogously, $q_{a,t}^{\text{fwd}}$ represents forward flow while $q_{a,t}^{\text{bwd}}$ represents backward flow.

2.5 Artificial compressor arcs

Finally, there are the compressor arcs, which are the only elements able to increase the gas pressure in the direction of the flow and for which we create a new configuration-based model in the following. Compressor arcs approximate the capabilities of compressor station arcs in the original network station topology. For this reason, we shortly introduce these in the following.

Each compressor station, which we assume to be modeled as an arc $a = (\ell, r)$ as well, represents a combination of single compressor units, which again are a combination of a compressor machine and a corresponding drive that provides the necessary power. The compressor machine of a compressor unit u can operate in its feasible operating range, we assume to be given as a polytope in the space (Q, ψ) of the volumetric flow Q and the compression ratio ψ given as

$$p = \rho R_s T_a z_a, \quad (9a)$$

$$Q = \frac{q}{\rho \ell} \implies Q = q \frac{R_s T_a z_a}{p \ell}, \quad (9b)$$

$$\psi = \frac{p_r}{p_\ell}. \quad (9c)$$

Here, ρ denotes the density of the gas, ρ_ℓ the incoming gas density, while z_a and T_a are defined in the same way as they were for pipes in Section 2.2. Equation (9a) is called the equation of state for real gases [35, 13]. We note that in reality, the feasible operating range is a non-linear set as, for example, described in [43], where also possible transformations to a polyhedral formulation are discussed.

The compression is additionally restricted by the maximal amount of power \bar{P}_u the corresponding drive can provide. The power needed for compressing a mass flow value of q from p_ℓ to p_r , i.e., with $\psi = p_r/p_\ell$, is given as

$$P(q, p_\ell, p_r) = \frac{q}{\eta_{\text{ad}}} R_s T_a z_a \frac{\kappa}{\kappa - 1} \left[\psi^{\frac{\kappa-1}{\kappa}} - 1 \right]. \quad (10)$$

Here, η_{ad} is the adiabatic efficiency and is assumed to be a constant given per compressor unit. Further, κ is the isentropic exponent, which we approximate by the constant value of 1.296, see, for example, [13] for more details. The single units can then be combined to predefined configurations \mathcal{C}_a of the compressor station a , defined as serial connections of parallel unit arrangements. For a more in-depth description, we refer to [13, 43, 22].

In the base model of [25], the artificial pressure-increasing arcs were modeled as a single compressor representing the combination of single units, which could dynamically be assigned to pressure-increasing arcs using binary assignment variables. As a consequence of this dynamic combination of different units, some of the constraints describing the resulting compression capabilities were approximated quite generously to keep the overall model linear.

To avoid this problem, we develop a novel formulation, which is closer to the original compressor station model and does not increase the model size in terms of variable and constraint numbers for real-world instances of the problem as shown in the results Section 4.1.1. As for the compressor stations, we create for each artificial compressor arc $a = (\ell, r) \in \mathcal{A}^{\text{arCp}}$ a set of configurations $\mathcal{C}_a^{\text{ar}}$ to choose from. These represent different compression settings based on how the compressor units can be composed in the underlying compressor stations. The feasible operating range of a chosen configuration $c \in \mathcal{C}_a^{\text{ar}}$ of an artificial compressor arc a is restricted in terms of three parameters: The maximum compression ratio ψ_c , the maximum volumetric flow Q_c , and the maximum compression power P_c . Furthermore, each configuration has an associated number u_c of used compressor units. From the parameters restricting the feasible operating range, we further find parameters $\alpha_c^{\text{p}\ell}$, $\alpha_c^{\text{p}r}$, and α_c^{q} , which represent a linearization of the compression power as

$$P \approx \alpha_c^{\text{p}\ell} p_\ell + \alpha_c^{\text{p}r} p_r + \alpha_c^{\text{q}} q,$$

and are determined by a linear regression based on triples $(p_\ell^s, p_r^s, q^s)_{s \in \mathbb{N}}$ sampled from within the intersection of the corresponding variable bound intervals over time, which satisfy the constraints

$$q^s \frac{R_s T_a z_a}{p_\ell^s} \leq Q_c, \quad \frac{p_r^s}{p_\ell^s} \leq \psi_c, \quad P(q^s, p_\ell^s, p_r^s) = P_c.$$

The final component of the configuration-based model is the set of artificial compressor configuration conflicts \mathcal{Z} . Each conflict z is defined as tuple $z = (c_1, c_2)$ of configurations of different artificial compressors from the same network station that cannot be used simultaneously. Conflicts are needed to model, for example, compressor units that can be used in different compressor stations but not at the same time. As a consequence, configurations c_1 and c_2 of two different artificial compressors cannot be used simultaneously if both incorporate a common compressor unit. Since conflicts are always defined for configurations of artificial compressors in the same network station, we can partition the conflicts per network station as $\mathcal{Z} = \bigcup_{i \in \mathcal{I}} \mathcal{Z}_i$.

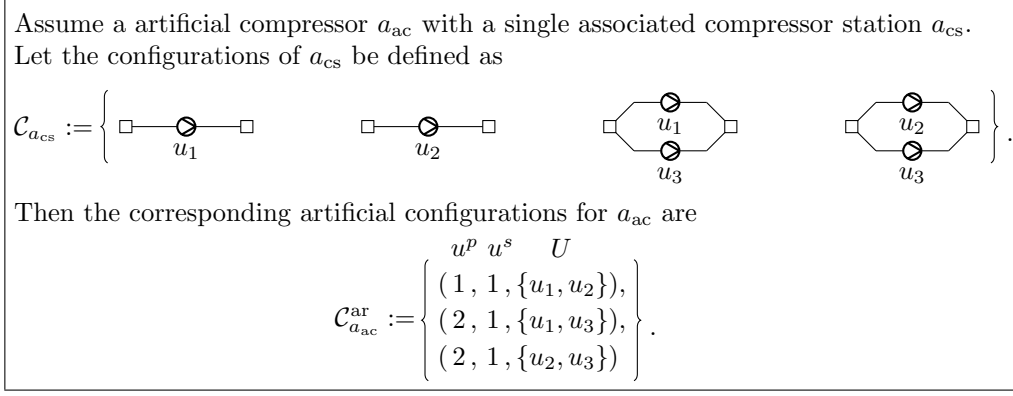
Given this description, we state the configuration-based model for an artificial compressor $a = (\ell, r) \in \mathcal{A}^{\text{arCp}}$ and time $t \in \mathcal{T}$ as

$$x_{a,t}^{\text{arc}} = \sum_{c \in \mathcal{C}_a^{\text{ar}}} x_{c,t}^{\text{cfg}}, \quad (11a)$$

$$x_{a,t}^{\text{arc}} = 1 \implies p_{\ell,t} \leq p_{r,t}, \quad (11b)$$

$$x_{a,t}^{\text{arc}} = 0 \implies q_{a,t} \leq 0, \quad (11c)$$

$$q_{a,t} \geq 0, \quad (11d)$$



■ **Figure 1** Example of the configuration construction for an artificial compressor a_{ac} . The associated compressor station a_{cs} has four different configurations using three different compressor units: two configurations using only a single unit, either unit u_1 or unit u_2 , and two configurations in which each of the two units is used in parallel to unit u_3 . There are three resulting artificial configurations for a_{ac} , which all allow only one serial stage: one configuration with only one allowed parallel unit, in which u_1 or u_2 can be chosen, and two configurations allowing for two parallel units, one with the units set $\{u_1, u_3\}$ and one with the set $\{u_2, u_3\}$. While we can combine the first two original configurations into one artificial configuration, this is not possible for the parallel configurations. The corresponding unit set would have to be $\{u_1, u_2, u_3\}$, which is not feasible as there is no original configuration combining u_1 and u_2 in parallel. Figure created with TikZ [40].

where we additionally demand for each configuration $c \in \mathcal{C}_a^{\text{ar}}$ of $a = (\ell, r) \in \mathcal{A}^{\text{arCp}}$ that

$$x_{c,t}^{\text{cfg}} = 1 \implies p_{r,t} \leq p_{\ell,t} \psi_c \quad \forall t \in \mathcal{T}, \quad (12a)$$

$$x_{c,t}^{\text{cfg}} = 1 \implies q_{a,t} \leq p_{\ell,t} \frac{Q_c}{R_s T_a z_a} \quad \forall t \in \mathcal{T}, \quad (12b)$$

$$x_{c,t}^{\text{cfg}} = 1 \implies \alpha_c^{\text{p}\ell} p_{\ell,t} + \alpha_c^{\text{p}r} p_{r,t} + \alpha_c^{\text{q}} q_{a,t} \leq P_c \quad \forall t \in \mathcal{T}, \quad (12c)$$

$$x_{c,t}^{\text{cfg}} \in \{0, 1\} \quad \forall t \in \mathcal{T}_0.$$

Here, the binary variable $x_{c,t}^{\text{cfg}}$ represents the decision to use (value=1) or not to use (value=0) the corresponding configuration $c \in \mathcal{C}_a^{\text{ar}}$ at time t . In addition, we need the following constraint for each conflict $z = (c_1, c_2) \in \mathcal{Z}$ and time $t \in \mathcal{T}$:

$$x_{c_1,t}^{\text{cfg}} + x_{c_2,t}^{\text{cfg}} \leq 1. \quad (13)$$

Note that the artificial compressor arcs are directed and therefore can only have positive flow as well as compress gas from ℓ to r , which is ensured via Equations (11d) and (11b).

2.5.1 Artificial configuration set construction

Since the artificial compressor configurations are not given as part of the input, we create them for each artificial compressor $a \in \mathcal{A}^{\text{arCp}}$ based on the original compressor stations represented by it. Here, we associate each artificial configuration c with a tuple (u_c^p, u_c^s, U_c) , where u_c^p is the number of parallel compressor units to use, u_c^s is the number of serial units to use, and U_c is the set of usable units. We manually create these tuples based on the configurations of the original compressor stations as well as their topological position in the network. If multiple original compressor compositions use similar units in the same layout, we represent them by the same artificial configuration if possible. An example of the construction is given in Figure 1. Note that the aggregation of all compressor configurations using the same layout to a single artificial configuration may not be possible if some of the involved units can be used in multiple compressor stations.

Given (u_c^p, u_c^s, U_c) for each artificial configuration c , we now derive the corresponding parameters for the number of simultaneously used compressor units u_c , the maximum compression ratio ψ_c , the maximum volumetric flow Q_c , and the maximum power P_c . First, we present a procedure for a configuration c for the special case of either u_c^p or u_c^s being equal to 1, i.e., for the case of pure serial or pure parallel compression.

For each compressor unit $u \in U_c$, we are given the maximum power \bar{P}_u its associated drive can provide. Furthermore, we can determine its maximum compression ratio $\bar{\psi}_u$ and maximum volumetric flow \bar{Q}_u by iterating

the extreme points of its feasible operating range polytope. Let then $(\tilde{\psi}_n)_{n=1}^{|U_c|}$ be the sequence of maximum compression ratios $\tilde{\psi}_u$ for all $u \in U_c$ sorted in descending order, and analogously let $(\tilde{Q}_n)_{n=1}^{|U_c|}$ and $(\tilde{P}_n)_{n=1}^{|U_c|}$ be the corresponding descending sequences of maximum volumetric flows \tilde{Q}_u and power values \tilde{P}_u . We then define the parameters u_c , P_c , ψ_c , and Q_c for $c \in \mathcal{C}_a^{\text{ar}}$ of $a \in \mathcal{A}^{\text{arCP}}$ as

$$u_c = u_c^p \cdot u_c^s, \quad P_c = \sum_{i=1}^{u_c} \tilde{P}_i, \quad (14a)$$

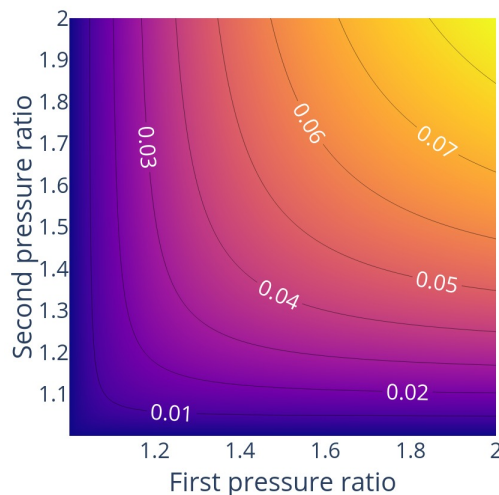
$$\psi_c = \begin{cases} \prod_{i=1}^{u_c^s} \tilde{\psi}_i & \text{if } u_c^s > 1 \\ \tilde{\psi}_{u_c^p} & \text{if } u_c^s = 1 \end{cases}, \quad Q_c = \begin{cases} \sum_{i=1}^{u_c^p} \tilde{Q}_i & \text{if } u_c^p > 1 \\ \tilde{Q}_{u_c^s} & \text{if } u_c^p = 1 \end{cases}. \quad (14b)$$

Note that for the serial compression, summing up the power values is not quite accurate and slightly underestimates the necessary compression power. As a consequence, P_c is potentially too small to allow for all compression processes possible in the original network. In Figure 2, we compare the total power P_{sum} needed for a serial compression of two units having the compression ratios ψ_1 and ψ_2 with the power P_{one} necessary for the combined compression process, in which a single unit is used with a compression ratio of $\psi_1 \cdot \psi_2$. For a given combination of ψ_1 and ψ_2 , we plot the relative difference $(P_{\text{one}} - P_{\text{sum}})/P_{\text{sum}}$. In the data used for our computational experiments in Section 4, the maximum serial compression number u_c^s for a configuration c is 2. In addition, all involved units have compressor ratios smaller than 2.0. Thus, we can deduce from Figure 2 that the approximation error is guaranteed to be smaller than 9%.

If a configuration features multiple serial and multiple parallel stages, the above-presented formulas for creating the configuration parameters cannot be applied directly. Instead, we create for each such arrangement and each concrete assignment of a compressor unit to a position in the arrangement a separate artificial configuration for the corresponding artificial compressor arc. The total number of used compressor units u_c is then equal to the number of active units in each specific arrangement, and the maximum power P_c is determined as the sum of maximum power values of these units. To determine the parameters ψ_c and Q_c , we recursively apply the formulas (14b) for each of the concrete serial and parallel subcompositions of units. The approach of creating one configuration for each possible unit arrangement can, in theory, result in a lot of artificial configurations. However, in our experience, this is not the case in practice, as such configurations do not occur very often.

2.6 Station simple states

The stations' artificial arcs do not operate independently from each other. Instead, each station $i \in \mathcal{I}$ is always in exactly one *simple state*, where the set of possible simple states of the station is given as \mathcal{S}_i , and the union



■ **Figure 2** Relative error in compression power. We compare the power P_{sum} needed for a serial compression using two separate compression processes with ratios ψ_1 and ψ_2 to the power P_{one} needed by our approximation of using a single compression process with ratio $\psi_1 \cdot \psi_2$. We plotted the relative error $(P_{\text{one}} - P_{\text{sum}})/P_{\text{sum}}$ for compression ratios ψ_1 and ψ_2 up to 2.0. Figure created with [38].

of simple states over all stations is given as $\mathcal{S} = \bigcup_{i \in \mathcal{I}} \mathcal{S}_i$. Each simple state $s \in \mathcal{S}_i$ restricts the usage of the artificial arcs by defining a set of arcs $\mathcal{A}_s^{\text{on}} \subseteq \mathcal{A}_i^{\text{ar}}$, which need to be active if the station uses state s , and a set of arcs $\mathcal{A}_s^{\text{off}} \subseteq \mathcal{A}_i^{\text{ar}}$, which cannot be active if the station uses state s . For all other artificial arcs $\mathcal{A}_i^{\text{ar}} \setminus (\mathcal{A}_s^{\text{on}} \cup \mathcal{A}_s^{\text{off}})$, the activity is optional. We represent the choice for a simple state s at time t by the binary variable $x_{s,t}^{\text{st}}$, where a value of 1 represents an active simple state. Using this, we formulate the simple state model for each network station $i \in \mathcal{I}$ as

$$\begin{aligned} \sum_{s \in \mathcal{S}_i} x_{s,t}^{\text{st}} &= 1 & \forall t \in \mathcal{T}, \\ x_{s,t}^{\text{st}} &\in \{0, 1\} & \forall s \in \mathcal{S}_i \quad \forall t \in \mathcal{T}_0, \end{aligned} \quad (15)$$

and for all artificial arcs $a \in \mathcal{A}_i^{\text{ar}}$ of each network station $i \in \mathcal{I}$ and time $t \in \mathcal{T}$ as

$$\sum_{s \in \mathcal{S}_i: a \in \mathcal{A}_s^{\text{on}}} x_{s,t}^{\text{st}} \leq x_{a,t}^{\text{arc}}, \quad 1 - \sum_{s \in \mathcal{S}_i: a \in \mathcal{A}_s^{\text{off}}} x_{s,t}^{\text{st}} \geq x_{a,t}^{\text{arc}}. \quad (16)$$

2.7 Station flow directions and fence node valves

Apart from the simple state, each station is also assigned exactly one *flow direction* f at each point in time $t \in \mathcal{T}_0$, captured by the binary variable $x_{f,t}^{\text{fd}}$, where the value 1 again stands for an active flow direction. The set of possible flow directions for station $i \in \mathcal{I}$ is given as \mathcal{F}_i . A flow direction $f \in \mathcal{F}_i$ determines the connection between the interior and exterior of each station by defining two sets of fence nodes: those fence nodes $\mathcal{V}_f^{\text{in}} \subseteq \mathcal{V}_i^{\text{fn}}$ allowing for flow into the station and those fence nodes $\mathcal{V}_f^{\text{out}} \subseteq \mathcal{V}_i^{\text{fn}}$ allowing for flow out of the station. All other fence nodes $\mathcal{V}_i^{\text{fn}} \setminus (\mathcal{V}_f^{\text{in}} \cup \mathcal{V}_f^{\text{out}})$ do not allow any exchange of flow between the station's interior and exterior.

As an extension to the model given in [25], we also decouple the pressure at those fence nodes without flow exchange. To do that, we adopt an idea initially mentioned in [24]: We create a copy v' of each fence node v , create a valve a' in between the two nodes, and reconnect each artificial arc that was connected to the fence node v to its copy v' instead. Note that we add the copy v' to \mathcal{V} and the new valve a to \mathcal{A}^{va} . Thereby, we create all the corresponding variables and constraints. We further denote by a_v^{vaFn} the *fence node valve* of fence node v , by $\mathcal{A}^{\text{vaFn}} \subseteq \mathcal{A}^{\text{va}}$ the set of all fence node valves contained in the network, and by $\mathcal{A}^{\text{vaOr}} = \mathcal{A}^{\text{va}} \setminus \mathcal{A}^{\text{vaFn}}$ the set of original non-fence-node valves of the network.

In addition to the flow restrictions, some fence nodes v have an upper pressure bound \bar{p}_v^{exit} in case flow leaves the station over v for some flow direction, i.e., the fence node serves as an exit of the station. For all other fence nodes v , we set $\bar{p}_v^{\text{exit}} = \bar{p}_v$.

Finally, not all flow directions fit to all simple states. Hence, we are given the set $\mathcal{Q} \subset \mathcal{F} \times \mathcal{S}$ of suitable combinations of flow directions and simple states.

For the model regarding flow directions and fence node valves, we demand for all network stations $i \in \mathcal{I}$ and all times $t \in \mathcal{T}$ that

$$\begin{aligned} \sum_{f \in \mathcal{F}_i} x_{f,t}^{\text{fd}} &= 1, \\ x_{f,t}^{\text{fd}} &\in \{0, 1\} \quad \forall f \in \mathcal{F}_i, \end{aligned} \quad (17)$$

for all simple states $s \in \mathcal{S}_i$ of all stations $i \in \mathcal{I}$ and all times $t \in \mathcal{T}$ that

$$\sum_{(f,s) \in \mathcal{Q}} x_{f,t}^{\text{fd}} \geq x_{s,t}^{\text{st}}, \quad (18)$$

and finally, for all fence nodes $v \in \mathcal{V}_i^{\text{fn}}$ of all stations $i \in \mathcal{I}$ and all times $t \in \mathcal{T}$ that

$$\sum_{f \in \mathcal{F}_i: v \in (\mathcal{V}_f^{\text{in}} \cup \mathcal{V}_f^{\text{out}})} x_{f,t}^{\text{fd}} = m_{a_v^{\text{vaFn}}, t}^{\text{op}}, \quad (19a)$$

$$\sum_{f \in \mathcal{F}_i: v \in \mathcal{V}_f^{\text{in}}} x_{f,t}^{\text{fd}} = 1 \implies q_{a_v^{\text{vaFn}}, t} \geq 0, \quad (19b)$$

$$\sum_{f \in \mathcal{F}_i: v \in \mathcal{V}_f^{\text{out}}} x_{f,t}^{\text{fd}} = 1 \implies q_{a_v^{\text{vaFn}}, t} \leq 0, \quad (19c)$$

$$\sum_{f \in \mathcal{F}_i: v \in \mathcal{V}_f^{\text{out}}} x_{f,t}^{\text{fd}} = 1, \implies p_{v,t} \leq \bar{p}_v^{\text{exit}}. \quad (19d)$$

2.8 Demand scenario

At each boundary node, we are given future requirements in terms of inflow and pressure, which we call the *demands*. A distinct set of demand values for all boundary nodes of the network is called a *demand scenario* of the problem. While for each future time point $t \in \mathcal{T}_0$, there is a required inflow value $\widehat{d}_{v,t}$ for each boundary node $v \in \mathcal{V}^b$, the prescribed pressure $\widehat{p}_{v,t}$ only exists for entries $v \in \mathcal{V}^+$ with non-zero inflow values at time t . Furthermore, we do not have to meet them exactly but within a tolerance of $\varepsilon = 1$ bar for each value.

However, there is no guarantee that meeting the demands is actually possible. For these cases, we allow a deviation from the demands, which we measure and penalize in the objective function. We call this deviation *slack* and capture it in the variables $\sigma_{v,t}^{d+}$ and $\sigma_{v,t}^{d-}$ for the positive and negative inflow deviation and $\sigma_{v,t}^{p+}$ and $\sigma_{v,t}^{p-}$ for the positive and negative pressure deviation for a boundary node $v \in \mathcal{V}^b$ and time $t \in \mathcal{T}$. When denoting the maximum amount of inflow slack per boundary node and time point by $\bar{\sigma}^d$ as well as the corresponding maximum amount of pressure slack by $\bar{\sigma}^p$, we can formulate the model of the inflow demands for all boundary nodes $v \in \mathcal{V}^b$ and a future time $t \in \mathcal{T}$ as

$$d_{v,t} = \widehat{d}_{v,t} + \sigma_{v,t}^{d+} - \sigma_{v,t}^{d-}, \quad (20a)$$

$$0 \leq (\sigma_{v,t}^{d+}, \sigma_{v,t}^{d-}) \leq \bar{\sigma}^d, \quad (20b)$$

and the model of the pressure demands for all entry nodes $v \in \mathcal{V}^+$ with non-zero inflow demands $\widehat{d}_{v,t}$ and a future time $t \in \mathcal{T}$ as

$$p_{v,t} \leq \widehat{p}_{v,t} + \varepsilon + \sigma_{v,t}^{p+}, \quad (21a)$$

$$p_{v,t} \geq \widehat{p}_{v,t} - \varepsilon - \sigma_{v,t}^{p-}, \quad (21b)$$

$$0 \leq (\sigma_{v,t}^{p+}, \sigma_{v,t}^{p-}) \leq \bar{\sigma}^p. \quad (21c)$$

2.9 Initial state of artificial elements

As stated at the end of Section 2.1, the initial state of the network specifies the pressures, flows, and modes of all nodes and elements of the original network. However, since the network station model is based on an artificial topology, there are no initial state values for the non-fence nodes inside the station, the artificial arcs, the station's simple state, and its flow direction.

For the initial pressures and temperatures of the inner station nodes, which are used as parameters in the artificial compressor model, we use the corresponding average values per quantity over the set of fence nodes. As the only exception, we use for the fence node copies the initial quantities given for the original fence nodes. Regarding the flow values, we need the initial flow on the fence node valves for the objective function (see Section 2.10), which can be determined by using the flow balance equation (1) of the corresponding fence node.

Regarding the discrete states of the network stations as well as their contained artificial arcs, we aim to choose settings that fit the initial state of the fence nodes, fence node valves, and the compressor stations in the original network station topology. Since there might be multiple suitable settings, we only sort out non-fitting ones and include the final decision into the optimization model.

First, we determine a list of possible initial simple states \mathcal{S}^0 . This is done by checking for each simple state of each network station if there would be a corresponding feasible initial state for the flow direction choice and all the elements in the network station, which fits to the initial pressure and flow values of the fence nodes and corresponding fence node valves. As a second step, we determine the set of initially running compressor units \mathcal{U}^0 based on the active configurations of the compressor stations in the original network station topology. Finally, we determine the set $\mathcal{C}^{\text{ar}0}$ of those configurations of artificial compressors c , for which there are enough initially running compressor units contained in their set of usable compressor units U_c , such that the configuration might be active.

Having these sets, we can define the model for the initial time point. From the above-defined constraints, we use (11a), (13), (15), and (16) also for $t = 0$. In addition, we add the following new constraints:

$$x_{s,0}^{\text{st}} = 0 \quad \forall s \in \mathcal{S} \setminus \mathcal{S}^0, \quad (22)$$

$$x_{c,0}^{\text{cfg}} = 0 \quad \forall a \in \mathcal{A}^{\text{arCp}} \quad \forall c \in \mathcal{C}_a^{\text{ar}} \setminus \mathcal{C}^{\text{ar}0}, \quad (23)$$

$$\sum_{a \in \mathcal{A}^{\text{arCp}}} \sum_{c \in \mathcal{C}_a^{\text{ar}}: u \in U_c} x_{c,0}^{\text{cfg}} \geq 1 \quad \forall u \in \mathcal{U}^0. \quad (24)$$

While the first two constraints ensure that only initially valid simple states and artificial configurations are chosen for the initial state, the last constraint demands for each initially running compressor unit u that there is at least one initially active artificial configuration c with $u \in U_c$.

2.10 Objective function

The general goal of the model is to reduce the amount of changes necessary to satisfy the given demands while trying to reduce unnecessary compressor unit usage. In contrast to the base model of [25], where a multi-level model was formulated to ensure each solution has minimal slack values, we instead used a single-level objective function representing the sum of differently weighted terms. Another difference from the base model is that we do not perform a solution *smoothing* in a post-processing step but instead include this already in the model, which increases its complexity.

To build our objective function, we need to introduce additional variables to track the corresponding changes of the single entities and connect these with the network state quantities. We start with the changes of valve modes. For each non-fence-node valve $a \in \mathcal{A}^{\text{vaOr}}$ and times $t_1 \in \mathcal{T}$ and $t_0 = t_1 - 1$, we capture a change in the binary variable $\delta_{a,t_1}^{\text{va}}$ and specify the corresponding model as

$$m_{a,t_1}^{\text{op}} - m_{a,t_0}^{\text{op}} \leq \delta_{a,t_1}^{\text{va}}, \quad (25a)$$

$$m_{a,t_0}^{\text{op}} - m_{a,t_1}^{\text{op}} \leq \delta_{a,t_1}^{\text{va}}, \quad (25b)$$

$$\delta_{a,t_1}^{\text{va}} \in \{0, 1\}.$$

For the regulator changes, we use the model introduced in [22] for tracking changes of the regulator's mode and, in case it is active, the regulator's operation point in terms of changes in the incoming pressure, outgoing pressure, and flow. For a regulator $a = (\ell, r) \in \mathcal{A}^{\text{rg}}$ and times $t_1 \in \mathcal{T}$ and $t_0 = t_1 - 1$, the model reads as

$$\delta_{a,t_1}^{\text{rg}} \geq m_{a,t_1}^{\text{x}} - m_{a,t_0}^{\text{x}} \quad \forall \text{x} \in \{\text{cl}, \text{op}, \text{ac}\}, \quad (26a)$$

$$\delta_{a,t_1}^{\text{rg}} \leq 2 - m_{a,t_1}^{\text{x}} - m_{a,t_0}^{\text{x}} \quad \forall \text{x} \in \{\text{cl}, \text{op}, \text{ac}\}, \quad (26b)$$

$$y_{t_1} - y_{t_0} \leq \delta_{a,t_1}^{\text{rg-y}} + (m_{a,t_1}^{\text{op}} + m_{a,t_1}^{\text{cl}} + \delta_{a,t_1}^{\text{rg}})(\bar{y}_{t_1} - \underline{y}_{t_0}) \quad \forall y = y \in \{p_\ell, p_r, q_a\}, \quad (26c)$$

$$y_{t_0} - y_{t_1} \leq \delta_{a,t_1}^{\text{rg-y}} + (m_{a,t_1}^{\text{op}} + m_{a,t_1}^{\text{cl}} + \delta_{a,t_1}^{\text{rg}})(\bar{y}_{t_0} - \underline{y}_{t_1}) \quad \forall y = y \in \{p_\ell, p_r, q_a\}, \quad (26d)$$

$$\delta_{a,t_1}^{\text{rg}} \in \{0, 1\},$$

where $\delta_{a,t_1}^{\text{rg}}$ denotes the binary variable indicating a change from one regulator mode to another one between times t_0 and t_1 , and $\delta_{a,t_1}^{\text{rg-pl}}$, $\delta_{a,t_1}^{\text{rg-pr}}$, and $\delta_{a,t_1}^{\text{rg-qa}}$ track the changes in the operation point for an active regulator for the incoming pressure, outgoing pressure, and flow through the element.

Next are the changes regarding the network stations and their corresponding elements. First, we introduce binary variables $\delta_{a,t}^{\text{arc}}$, representing a change in the activity of the artificial arc a between time t and the previous time point. The corresponding constraints read for an artificial arc $a \in \mathcal{A}^{\text{ar}}$ and times $t_1 \in \mathcal{T}$ and $t_0 = t_1 - 1$ as

$$x_{a,t_1}^{\text{arc}} - x_{a,t_0}^{\text{arc}} \leq \delta_{a,t_1}^{\text{arc}}, \quad (27a)$$

$$x_{a,t_0}^{\text{arc}} - x_{a,t_1}^{\text{arc}} \leq \delta_{a,t_1}^{\text{arc}}, \quad (27b)$$

$$\delta_{a,t_1}^{\text{arc}} \in \{0, 1\}.$$

For artificial compressors, we also count configuration changes as changes in the arc's activity. We only track if a configuration is newly activated, as turning it off either turns another configuration on or deactivates the whole compressor arc. The constraint for each configuration $c \in \mathcal{C}_a^{\text{ar}}$ of each artificial compressor $a \in \mathcal{A}^{\text{arCp}}$ and times $t_1 \in \mathcal{T}$ and $t_0 = t_1 - 1$ is then given as

$$x_{c,t_1}^{\text{cfg}} - x_{c,t_0}^{\text{cfg}} \leq \delta_{a,t_1}^{\text{arc}}. \quad (28)$$

As in the base model, we track changes in the network stations' simple states but not in the flow directions. Since each station has exactly one simple state at each point in time, we capture the activation of a simple state s at time t in the binary variable $\delta_{s,t}^{\text{st}}$ and give the model for a simple state $s \in \mathcal{S}_i$ of a station $i \in \mathcal{I}$ and times $t_1 \in \mathcal{T}$ and $t_0 = t_1 - 1$ as

$$x_{s,t_1}^{\text{st}} - x_{s,t_0}^{\text{st}} \leq \delta_{s,t_1}^{\text{st}}, \quad (29)$$

$$\delta_{s,t_1}^{\text{st}} \in \{0, 1\}.$$

Finally, we capture the pressure changes at each fence node v between time t and the previous time point in the variable $\delta_{v,t}^{\text{fn-p}}$ and the flow changes at the corresponding fence node valve by the variable $\delta_{v,t}^{\text{fn-q}}$. The model for a fence node $v \in \mathcal{V}_i^{\text{fn}}$ of network station $i \in \mathcal{I}$ and times $t_1 \in \mathcal{T}$ and $t_0 = t_1 - 1$ is given as

$$p_{v,t_1} - p_{v,t_0} \leq \delta_{v,t_1}^{\text{fn-p}}, \quad (30a)$$

$$p_{v,t_0} - p_{v,t_1} \leq \delta_{v,t_1}^{\text{fn-p}}, \quad (30b)$$

$$q_{a_v^{\text{vaFn}},t_1} - q_{a_v^{\text{vaFn}},t_0} \leq \delta_{v,t_1}^{\text{fn-q}}, \quad (30c)$$

$$q_{a_v^{\text{vaFn}},t_0} - q_{a_v^{\text{vaFn}},t_1} \leq \delta_{v,t_1}^{\text{fn-q}}. \quad (30d)$$

Having defined all the change-tracking variables, we are now able to formulate the objective function as a weighted sum of different penalty terms. Apart from the changes and slack values defined above, we also penalize the usage of compressor units by penalizing the usage of artificial configurations c weighted by their number of used compressor units u_c . For each of the terms in the sum, we have specific weights defining the penalty costs: the demand slack cost per hour $w^{\sigma\text{-d}}$, the pressure slack cost per hour $w^{\sigma\text{-p}}$, the cost for each compressor unit running in an active configuration per hour w^{ru} , the non-fence-node valve change cost w^{va} , the regulator mode change cost w^{rg} , the operation point change costs of active regulators for the incoming pressure $w^{\text{rg-pl}}$, the outgoing pressure $w^{\text{rg-pr}}$ and the flow $w^{\text{rg-q}}$, the artificial arc change costs w^{arc} , the costs for activating a simple state w_s^{st} , which are specific values manually defined by the industry experts for each simple state $s \in \mathcal{S}$, and the costs for changes of the fence node pressure $w^{\text{fn-p}}$ and flow changes on the corresponding fence node valves $w^{\text{fn-q}}$. The complete objective is then given as

$$\begin{aligned} \min \sum_{t \in \mathcal{T}} \left(\sum_{v \in \mathcal{V}^b} \frac{\tau(t) - \tau(t-1)}{1 \text{ h}} (w^{\sigma\text{-d}}(\sigma_{v,t}^{d+} + \sigma_{v,t}^{d-}) + w^{\sigma\text{-p}}(\sigma_{v,t}^{p+} + \sigma_{v,t}^{p-})) \right. \\ + \sum_{a \in \mathcal{A}^{\text{arCp}}} \sum_{c \in \mathcal{C}_a^{\text{ar}}} \frac{\tau(t) - \tau(t-1)}{1 \text{ h}} \cdot u_c \cdot w^{\text{ru}} \cdot x_{c,t}^{\text{cfg}} \\ + \sum_{a \in \mathcal{A}^{\text{vaOr}}} w^{\text{va}} \cdot \delta_{a,t}^{\text{va}} + \sum_{a \in \mathcal{A}^{\text{ar}}} w^{\text{arc}} \cdot \delta_{a,t}^{\text{arc}} + \sum_{i \in \mathcal{I}} \sum_{s \in \mathcal{S}_i} w_s^{\text{st}} \cdot \delta_{s,t}^{\text{st}} \\ + \sum_{a \in \mathcal{A}^{\text{rg}}} \left(w^{\text{rg}} \cdot \delta_{a,t}^{\text{rg}} + w^{\text{rg-pl}} \cdot \delta_{a,t}^{\text{rg-pl}} + w^{\text{rg-pr}} \cdot \delta_{a,t}^{\text{rg-pr}} + w^{\text{rg-q}} \cdot \delta_{a,t}^{\text{rg-qa}} \right) \\ \left. + \sum_{i \in \mathcal{I}} \sum_{v \in \mathcal{V}_i^{\text{fn}}} \left(w^{\text{fn-p}} \cdot \delta_{v,t}^{\text{fn-p}} + w^{\text{fn-q}} \cdot \delta_{v,t}^{\text{fn-q}} \right) \right). \end{aligned} \quad (31)$$

2.11 Complete model

By combining all the above-defined variables and constraints with the objective function, we can formulate our gas flow problem as the following MINLP model \mathcal{P} :

$$\begin{aligned} \min \quad & (31) \\ \text{s.t.} \quad & \forall t \in \mathcal{T} \\ & (1) \quad \forall v \in \mathcal{V}, \quad (2) \quad \forall a \in \mathcal{A}^{\text{pi}}, \\ & (3) \quad \forall a \in \mathcal{A}^{\text{va}}, \quad (4), (26) \quad \forall a \in \mathcal{A}^{\text{rg}}, \\ & (6) \quad \forall a \in \mathcal{A}^{\text{arSc}}, \quad (7) \quad \forall a \in \mathcal{A}^{\text{arRg}}, \\ & (8) \quad \forall a \in \mathcal{A}^{\text{arBiRg}}, \quad (11) \quad \forall a \in \mathcal{A}^{\text{arCp}}, \\ & (12), (28) \quad \forall a \in \mathcal{A}^{\text{arCp}} \quad \forall c \in \mathcal{C}_a^{\text{ar}}, \quad (13) \quad \forall z \in \mathcal{Z}, \\ & (15), (17) \quad \forall i \in \mathcal{I}, \quad (16), (27) \quad \forall i \in \mathcal{I} \quad \forall a \in \mathcal{A}_i^{\text{ar}}, \\ & (18), (29) \quad \forall i \in \mathcal{I} \quad \forall s \in \mathcal{S}_i, \quad (19), (30) \quad \forall i \in \mathcal{I} \quad \forall v \in \mathcal{V}_i^{\text{fn}}, \\ & (20) \quad \forall v \in \mathcal{V}^b, \quad (21) \quad \forall v \in \mathcal{V}^+ : \hat{d}_{v,t} \neq 0, \\ & (25) \quad \forall a \in \mathcal{A}^{\text{vaOr}}, \end{aligned}$$

$$\begin{aligned}
& t = 0 \\
\mathcal{P} \quad & (11a) \quad \forall a \in \mathcal{A}^{\text{arCp}}, & (13) \quad \forall z \in \mathcal{Z}, \\
& (15) \quad \forall i \in \mathcal{I}, & (16) \quad \forall i \in \mathcal{I} \quad \forall a \in \mathcal{A}_i^{\text{ar}}, \\
& (22) \quad \forall s \in \mathcal{S} \setminus \mathcal{S}^0, & (23) \quad \forall a \in \mathcal{A}^{\text{arCp}} \quad \forall c \in \mathcal{C}_a^{\text{ar}} \setminus \mathcal{C}^{\text{ar}0}, \\
& (24) \quad \forall u \in \mathcal{U}^0, \\
& \forall t \in \mathcal{T}_0 \\
& \{0, 1\} \ni m_{a,t}^{\text{op}} \quad \forall a \in \mathcal{A}^{\text{va}}, & m_{a,t}^{\text{op}}, m_{a,t}^{\text{ac}}, m_{a,t}^{\text{cl}} \quad \forall a \in \mathcal{A}^{\text{rg}}, \\
& x_{a,t}^{\text{arc}} \quad \forall a \in \mathcal{A}^{\text{ar}}, & x_{c,t}^{\text{cfg}} \quad \forall a \in \mathcal{A}^{\text{arCp}} \quad \forall c \in \mathcal{C}_a^{\text{ar}}, \\
& x_{s,t}^{\text{st}} \quad \forall i \in \mathcal{I} \quad \forall s \in \mathcal{S}_i, \\
& \forall t \in \mathcal{T} \\
& \{0, 1\} \ni x_{a,t}^{\text{fwd}}, x_{a,t}^{\text{bwd}} \quad \forall a \in \mathcal{A}^{\text{arBiRg}}, & x_{f,t}^{\text{fd}} \quad \forall i \in \mathcal{I} \quad \forall f \in \mathcal{F}_i, \\
& \delta_{a,t}^{\text{va}} \quad \forall a \in \mathcal{A}^{\text{vaOr}}, & \delta_{a,t}^{\text{rg}} \quad \forall a \in \mathcal{A}^{\text{rg}}, \\
& \delta_{a,t}^{\text{arc}} \quad \forall a \in \mathcal{A}^{\text{ar}}, & \delta_{s,t}^{\text{st}} \quad \forall s \in \mathcal{S}.
\end{aligned}$$

2.12 Model comparison

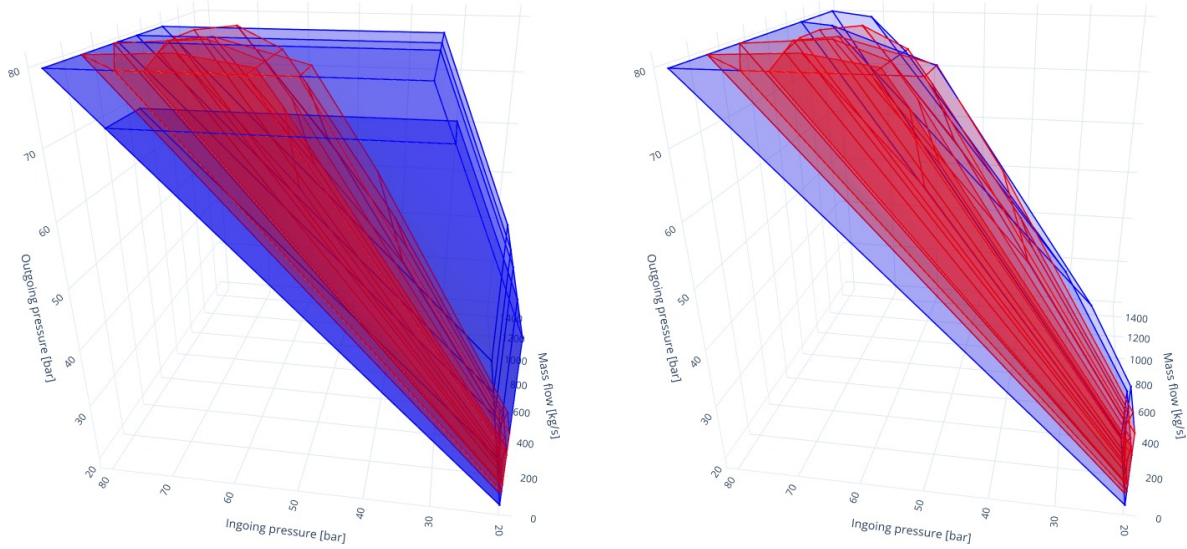
As final part of the model presentation, we summarize for the interested reader the differences between our problem formulation and the one of the base model of [25].

- As the most crucial change, we introduced in Section 2.5 a new model for the artificial compressor arcs in the network stations. It is based on the usage of configurations for each artificial arc, which we construct based on the properties of the compressor stations of the original network station topology.
- The second major difference is the introduction of artificial valves at the boundaries of the network stations in Section 2.7. With these, it is possible to decouple the pressure inside and outside the station at these points. Note that this was initially suggested in [24].
- In this paper, we use a single objective function representing the weighted sum of different terms to penalize them instead of a tri-level model, see Section 2.10. This objective also includes *smoothing* terms, which minimize pressure and inflow changes at the network station boundaries over time and an incentive to reduce the usage of compressor units to avoid unnecessary energy consumption.
- We use the regulator model of [22], which is slightly different from the base regulator model.
- We do not include artificial arc types that do not appear in our real-world-based test instances in Section 4.1. Hence, there are no *combined arcs*, which have gas compression and regulation capabilities, no *bi-directional compressor arcs* and no *bi-directional combined arcs*. Moreover, we do not support *flow direction conditions* for the same reason.

The main benefit of the new artificial compressor model introduced in Section 2.5 in the Equations (11), (12), and (13) compared to the base model is that each compressor unit contributes either in serial or in parallel to the compression capabilities and does not use both benefits at the same time.

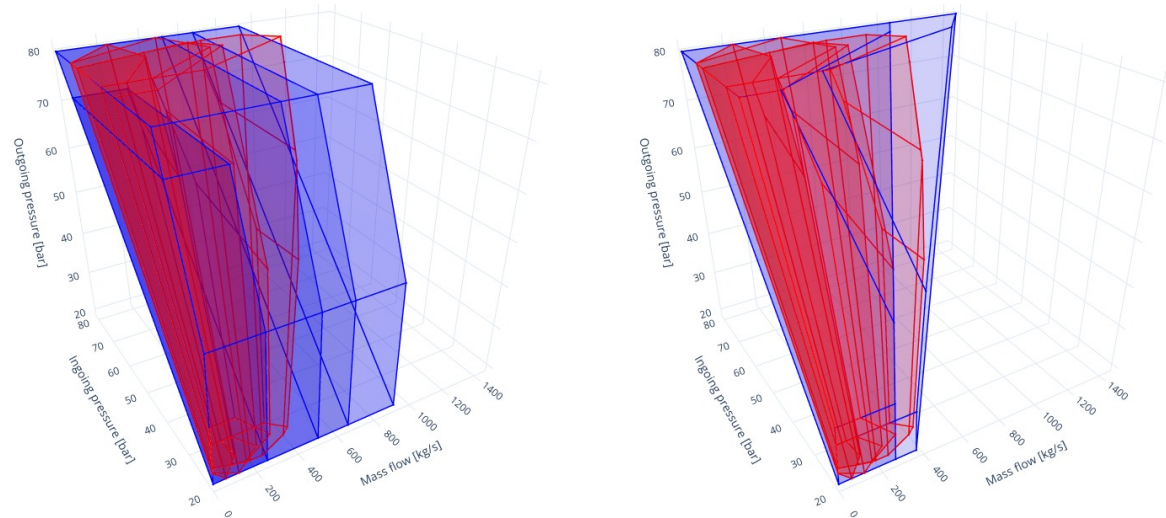
Furthermore, we improved the description of the maximum compression ratio ψ and the maximum volumetric flow Q . For both, we do not need to replace the incoming pressure of the arc p_ℓ by a constant value, which was done explicitly in Equation (40) in [25] and implicitly by assuming a maximum mass flow bound per compressor unit, which can only be derived from the inherent volumetric flow bound by assuming a constant incoming pressure. Moreover, we can correctly employ the product of compressor ratios in the computation of the maximum usable compression ratio of serial compression processes in the first of the formulas (14b) instead of the linear approximation formula used in (35) in [25].

As an example, we show the model improvements in comparison to [25] for a single artificial compressor arc in Figure 3, where we compare the polytopes representing the feasible region of the base model, the ones for our new formulation, and the actual feasible region of the configuration polytopes of the compressor station in the original station topology. To have a clean comparison, we choose an artificial compressor arc representing exactly one compressor station. This compressor station features three compressor units, from which two units can be combined in parallel, but has no serial unit combination capabilities. Note that we excluded the power bound from the polytopes since it is modeled for both models in the same way, i.e., as a linear approximation using a linear regression approach.



(a) Base model, View 1

(b) New model, View 1



(c) Base model, View 2

(d) New model, View 2

Figure 3 Feasible region comparison of the artificial compressor base model and our new one at the example of one artificial compressor arc. As a reference solution, we plotted the polytopes representing the configurations’ feasible regions of the associated compressor station from the original network topology in red. The polytopes for the artificial compressor arc are plotted in blue, where Figures (a) and (c) show the base model polytopes and Figures (b) and (d) show the new model polytopes. For the new model, each polytope represents one artificial configuration to choose from. Since the base model does not feature configurations, we plotted one polytope for each feasible compressor unit combination on the artificial arc. The pressure bound interval used for both endnodes is [20 bar, 80 bar]. For the constant incoming pressure needed for the base model, we use the median value of 50 bar. For a better impression of the three-dimensional polytopes, we show them from two different camera positions in View 1, visible in Figures (a) and (b), and View 2, visible in Figures (c) and (d). Figures created with [38].

We clearly see the increased accuracy of the new model already for an artificial compressor arc, which is only able to combine two compressor units. When combining more compressor units, the differences would further increase. However, we stress that despite the improved accuracy also the new model is only an approximation and no relaxation of the actual compression capabilities.

3 MIP-based heuristic algorithms

We aim at finding solutions for the transient gas flow model \mathcal{P} presented in the previous section for challenging demand scenarios on large instances based on real-world data. Since preliminary experiments showed that this is currently out of reach for general-purpose MINLP solvers, we created a problem-specific heuristic, which is described in Algorithm 1. It is based on the idea of dealing with the two sources of complexity, the non-continuity of binary variables and the non-linearity of the constraints, separately: In the first step, we determine multiple promising and feasible sets of solution values for all the binary variables, which we call *binary assignments* for the problem. Afterward, we transform the problem by fixing these values for each binary assignment and then solve the resulting NLP with a suitable general-purpose solver. Finally, we return the best obtained solution among all successful NLP runs. If none of the runs found a feasible solution, the heuristic failed. Note that the algorithm could easily be parallelized by solving the NLP model immediately after finding a new binary assignment and in parallel to the ongoing search for further binary assignments. Hence, feasible solutions would be produced early during the execution, which is especially valuable in environments with strict time limit restrictions.

Since we consider particularly challenging large instances, we purely focus on finding good feasible solutions fast, i.e., we do not aim for valid global lower bounds for the problem and thereby prove optimality of our solutions. However, note that in our approach, the general-purpose NLP solver usually provides, at least locally, optimal solutions for the corresponding sub-problem for a given binary assignment.

In the remainder of this section, we focus on finding binary assignments that yield high-quality solutions for the overall problem. Note that we sometimes claim that these binary-assignment-producing algorithms find solutions for the overall problem, which we use as a short form of explaining that we find solutions for the problem by using them as function `findBinaryAssignments` in Algorithm 1.

Algorithm 1: Base algorithm

Data: An instance Φ of model \mathcal{P}
Result: A valid solution for Φ

- 1 `baSet` \leftarrow `findBinaryAssignments`(Φ)
- 2 `solSet` \leftarrow `solveNLP`(`baSet`, Φ)
- 3 **return** `chooseBestSol`(`solSet`)

3.1 Sequential mixed-integer programming

The core of our algorithm for finding binary assignments consists of a sequential algorithm solving linearized versions of the original MINLP problem, which we refer to as **SMIP** and which is given as Algorithm 2. After initializing a linearization for the non-linear momentum equation (2b) in Line 1, we solve the remaining MIP problem. If successful, this yields a valid binary assignment in the sense that it respects all the constraints involving no continuous variables. We then check if the binary assignment was already found before and if the linearization error is small enough to satisfy a convergence criterion, which we describe in Section 3.1.1 below. If both are not the case and we have not reached the maximum number of iterations n^{\maxIt} , we update our linearization based on the current solution and repeat the process by solving the updated MIP model. The algorithm finally returns the set of found binary assignments to be completed to full solutions for the original model \mathcal{P} . Note that given a valid binary assignment, the resulting NLP model might be infeasible and hence does not yield a feasible overall solution.

To solve the single MIP problems, we use a general-purpose MIP solver, even though we are not interested in the provided lower bounds. Since primal heuristics are an essential and well-studied part of general-purpose MIP solvers that have a high priority early in the solving process [2], we expect to find good solutions quickly. Furthermore, we present in Section 3.3 a dynamic node limit to further accelerate this process. Nevertheless, it would also be possible to use other algorithms like heuristics to find good feasible solutions for the single MIP problems here.

Algorithm 2: General sequential mixed-integer programming algorithm (SMIP)**Data:** An instance Φ of model \mathcal{P} **Parameters:** Maximum number of iterations $n^{\max\text{It}}$ Absolute convergence tolerance $\varepsilon^{\text{tolAbs}}$ Relative convergence tolerance $\varepsilon^{\text{tolRel}}$ **Result:** A set of valid binary assignments for Φ

- 1 Create problem `mip` using the initial linearization.
- 2 **for** at most $n^{\max\text{It}}$ iterations **do**
- 3 Solve `mip` with a general-purpose solver. Abort if no primal solution was found.
- 4 Save the binary assignment `ba` of the solution.
- 5 Abort if `ba` was found before or the linearization error with respect to $\varepsilon^{\text{tolAbs}}$ and $\varepsilon^{\text{tolRel}}$ is small.
- 6 Update linearization of `mip` based on the current solution.
- 7 **return** all found binary assignments

3.1.1 Linearization and convergence

The only non-linear term in the model is the friction term in the momentum equation (2b) for pipes given as

$$c_a \cdot \left(\frac{|q_{\ell,a,t}|q_{\ell,a,t}}{p_{\ell,t}} + \frac{|q_{r,a,t}|q_{r,a,t}}{p_{r,t}} \right) \quad \text{with the constant } c_a := \frac{\lambda_a R_s T_a z_a L_a}{4A_a^2 D_a}.$$

Hence, we are given for each momentum equation of pipe $a = (\ell, r) \in \mathcal{A}^{\text{pi}}$ and time $t \in \mathcal{T}$ two non-linear functions of the form $c_a(|q_{x,a,t}|q_{x,a,t}/p_{x,t})$, one for each endnode $x \in \{\ell, r\}$. We approximate both functions by a linear combination of the involved quantities as

$$c_a \frac{|q_{x,a,t}|q_{x,a,t}}{p_{x,t}} \approx \alpha_{x,a,t}^0 + \alpha_{x,a,t}^p \cdot p_{x,t} + \alpha_{x,a,t}^q \cdot q_{x,a,t}.$$

To check if a solution of a linearized MIP model has converged to a non-linear solution, we use two tolerance parameters: the absolute convergence tolerance $\varepsilon^{\text{tolAbs}}$ and the relative convergence tolerance $\varepsilon^{\text{tolRel}}$. Based on them, we say that a solution with pressure values p' and mass flow values q' converged if for all pipes and future time points the differences between the linearized friction function and the non-linear friction function with respect to p' and q' are smaller than the tolerances. Hence, we check in Line 5 of Algorithm 2 if the following constraints are satisfied:

$$\begin{aligned} \forall t \in \mathcal{T} \quad \forall a = (\ell, r) \in \mathcal{A}^{\text{pi}}: \quad & \Lambda^{\text{lin}} = \alpha_{\ell,a,t}^0 + \alpha_{\ell,a,t}^p \cdot p'_{\ell,t} + \alpha_{\ell,a,t}^q \cdot q'_{\ell,a,t} \\ & \quad + \alpha_{r,a,t}^0 + \alpha_{r,a,t}^p \cdot p'_{r,t} + \alpha_{r,a,t}^q \cdot q'_{r,a,t}, \\ \Lambda^{\text{orig}} = & c_a \cdot \left(\frac{|q'_{\ell,a,t}|q'_{\ell,a,t}}{p'_{\ell,t}} + \frac{|q'_{r,a,t}|q'_{r,a,t}}{p'_{r,t}} \right), \\ \varepsilon^{\text{tolAbs}} \geq & |\Lambda^{\text{lin}} - \Lambda^{\text{orig}}|, \\ \text{and, if } |\Lambda^{\text{orig}}| > \varepsilon^{\text{zero}}, & \varepsilon^{\text{tolRel}} \geq \frac{|\Lambda^{\text{lin}} - \Lambda^{\text{orig}}|}{|\Lambda^{\text{orig}}|}. \end{aligned}$$

Here, the value of the linearized friction function with respect to the variable values of the given solution is denoted by Λ^{lin} and the corresponding non-linear friction function value by Λ^{orig} . Furthermore, we can only check for the relative tolerance if the non-linear friction value is not zero. We ensure this by comparing the term with a corresponding epsilon value $\varepsilon^{\text{zero}}$, which we define to be equal to 0.001 bar.

For the initial linearization determined in Line 1 of the SMIP Algorithm 2, we use the constant velocity approximation introduced by [20], which was also used as initial linearization in [25] and is given as

$$\alpha_{x,a,t}^0 = 0, \quad \alpha_{x,a,t}^p = 0, \quad \alpha_{x,a,t}^q = \frac{\lambda_a L_a}{4A_a D_a} |v_{x,a,0}|, \quad \text{with } |v_{x,a,0}| := \frac{R_s T_a z_a}{A_a} \frac{q_0}{p_{x,0}} \quad \text{and } q_0 := \max\{|q_{x,a,0}|, q^{\text{min}}\}.$$

Here, $|v_{x,a,0}|$ denotes the absolute value of the gas velocity based on the initial state quantities using a minimal absolute flow value of q^{min} to avoid $\alpha_{x,a,t}^q$ being zero. The advantage of this linearization is that it is an odd function with respect to q , i.e., $f(q) = -f(-q)$ holds, from which also $f(0) = 0$ follows.

For all subsequent iterations of Algorithm 2, we determine the linearization based on the pressure values p' and flow values q' of a previous MIP solution. Here, we use the tangential plane on the non-linear function in the point (p', q') as linearization, which is the first-order Taylor expansion at this point and is given as

$$\begin{aligned}\alpha_{x,a,t}^p &= c_a \cdot \frac{-|q'_{x,a,t}|q'_{x,a,t}}{(p'_{x,t})^2}, & \alpha_{x,a,t}^q &= c_a \cdot \frac{2|q'_{x,a,t}|}{p'_{x,t}}, \\ \alpha_{x,a,t}^0 &= c_a \cdot \frac{|q'_{x,a,t}|q'_{x,a,t}}{p'_{x,t}} - \alpha_{x,a,t}^p \cdot p'_{x,t} - \alpha_{x,a,t}^q \cdot q'_{x,a,t}.\end{aligned}$$

3.1.2 Comments on infeasibility

Algorithm 2 struggles with infeasibility in two different ways: First, the linearized MIP problems of each iteration might be infeasible even if the overall MINLP problem \mathcal{P} is not. Second, the algorithm has no functionality to detect the overall infeasibility of the general model \mathcal{P} .

Despite those structural weaknesses, we still expect the algorithm to perform well on real-world-based instances, like those of our computational experiments in Section 4. On these, infeasibility usually does not occur, as real-world gas transport networks are, in general, very flexible and allow for a broad range of different pressure, flow, and inflow values. In addition, the slack values can adjust problematic inflow and pressure demands.

The computational results we present in Section 4 confirm our expectations, as we found a feasible solution for each instance of the problem \mathcal{P} and rarely encountered infeasibility when solving the linearized MIP models, see Table 5.

3.2 Reduced time horizon heuristics

To further accelerate the solving process, we introduce additional heuristics that find binary assignments by solving a series of models on time horizons of reduced size, where the size of a time horizon is defined as the number of future time points. The heuristics are based on two well-known ideas for time-expanded problems: the rolling horizon and the aggregated horizon. While rolling horizon heuristics rely on iteratively solving the model for a subset of time steps that are gradually shifted to the future, the aggregated horizon combines certain time steps and afterward completes the aggregated solution to one for the whole time horizon.

Note that all of the heuristics introduced below still maintain the general sequential structure of Algorithm 2.

3.2.1 Rolling horizon heuristic

In our rolling horizon heuristic RH given in Algorithm 3, we only make a single change in contrast to the SMIP Algorithm 2: Instead of solving the linearized MIP for the whole time horizon in Line 3, we find a solution using a rolling horizon approach. It creates a reduced time horizon of fixed length n^{rolHor} , solves the corresponding reduced model, saves the solution of the first future time step, i.e., all corresponding variable values, and finally shifts the horizon one time step into the future for the next iteration. The reduced model is created by only considering variables or constraints for the time steps of the reduced horizon. As the initial state, we use the saved solution from the previous iteration and take the original initial state for the first iteration. Note that we still use the overall initial time point 0 to determine the model's constant parameters, like, for example, the fixed temperature T_a of a pipe $a \in \mathcal{A}^{\text{pi}}$. We repeat this procedure until we reach the iteration containing the very last future time point t^{max} , where the complete solution of the reduced model is saved.

As a minor detail, we note that for all variables without a fixed initial state for time $t = 0$, for example, for the simple state variables x^{st} , we use the model as defined in Section 2 for the first iteration and then add the there determined values for $t = 0$ to the corresponding solution state. For all subsequent iterations, we fix the values determined in the previous iteration for the initial state and can therefore ignore all the constraints added to the model for $t = 0$.

3.2.2 Aggregated horizon heuristic

For the aggregated horizon heuristic AH stated as Algorithm 4, we create a time horizon of reduced size n^{aggHor} by combining certain time steps, resulting in longer but fewer time steps in the overall horizon. We represent this by a function $\vartheta : \{0, 1, \dots, n^{\text{aggHor}}\} \rightarrow \mathcal{T}_0 = \{0, 1, \dots, t^{\text{max}}\}$, which maps the time points of the aggregated time horizon to time points of the original time horizon. A time interval between two subsequent time points t and

Algorithm 3: Sequential rolling horizon heuristic algorithm (RH)**Data:** An instance Φ of model \mathcal{P} **Parameters:** Size of the rolling horizon n^{rolHor} Maximum number of iterations n^{maxIt} Absolute convergence tolerance $\varepsilon^{\text{tolAbs}}$ Relative convergence tolerance $\varepsilon^{\text{tolRel}}$ **Result:** A list of valid binary assignments for Φ

```

1 Function rollingHorizonHeuristic(mip) is
2   Create reducedHorizon as the first  $n^{\text{rolHor}}$  future time steps of the time horizon of mip.
3   while True do
4     Create reducedMip from mip using only future time points from reducedHorizon and the last saved
       state as initial state. For the first iteration, the original initial state is used.
5     Solve reducedMip with a general-purpose solver. If no primal solution was found, return without a
       solution.
6     Saved the solution state of the first future time point.
7     If reducedHorizon contains the time point  $t^{\text{max}}$  save the solution of the remaining reduced horizon
       and abort the loop.
8     Shift reducedHorizon one time step into the future.
9   return The saved solution for all time points of the time horizon of mip.
10 return the result of SMIP applied to  $\Phi$  using  $n^{\text{maxIt}}$ ,  $\varepsilon^{\text{tolAbs}}$ , and  $\varepsilon^{\text{tolRel}}$ , when using
       rollingHorizonHeuristic to find solutions for mip instead of a general-purpose MIP solver

```

$t + 1$ in the aggregated horizon represents the combination of all original time intervals between the time points $\vartheta(t)$ and $\vartheta(t + 1)$. To combine a similar amount of original time steps for each time interval, we determine ϑ as

$$\bar{c} := \left\lceil \frac{t^{\text{max}}}{n^{\text{aggHor}}} \right\rceil, \quad c := \left\lfloor \frac{t^{\text{max}}}{n^{\text{aggHor}}} \right\rfloor \quad \text{larger and smaller number of original time intervals to combine,}$$

$$\bar{n} := t^{\text{max}} \bmod n^{\text{aggHor}} \quad \text{amount of larger aggregated time intervals,}$$

$$\vartheta(t) := \begin{cases} t \cdot \bar{c} & \text{if } t \leq \bar{n} \\ \bar{n} + t \cdot c & \text{if } \bar{n} < t \end{cases}.$$

An explicit example can be given as

$$t^{\text{max}} = 7 \quad \& \quad n^{\text{aggHor}} = 4 \quad \Rightarrow \quad \begin{array}{c|ccccc} t & 0 & 1 & 2 & 3 & 4 \\ \vartheta(t) & 0 & 2 & 4 & 6 & 7 \end{array}.$$

Note that $\vartheta(0) = 0$ and $\vartheta(n^{\text{aggHor}}) = t^{\text{max}}$ hold. If n^{aggHor} is a divisor of t^{max} , each aggregated time interval combines $\bar{c} = c$ original time intervals.

Using the function ϑ , we can now derive a time aggregated problem instance Φ^{agg} from the original problem instance Φ in Line 1 of Algorithm 4. Since all the element sets of Φ and Φ^{agg} coincide except for the time horizon, we are only missing a description of the time-dependent instance parameters. In a first step, we determine the inflow demands. As they represent the average inflow rate over a time interval, we determine the time aggregated inflow demands as the average of original inflow demands, weighted by the length of the corresponding time intervals, i.e., for the aggregated time $t \in \{1, \dots, n^{\text{aggHor}}\}$ and a boundary node $v \in \mathcal{V}^b$ we define

$$\hat{d}_{v,t}^{\text{agg}} := \frac{\sum_{i=\vartheta(t-1)+1}^{\vartheta(t)} (\tau(i) - \tau(i-1)) \cdot \hat{d}_{v,i}}{\tau(\vartheta(t)) - \tau(\vartheta(t-1))}.$$

For all the remaining time-dependent parameters, like, for example, the function $\tau(t)$ representing the time difference between t and the initial time point in seconds, the pressure demands at the entries, or the general pressure and flow bounds, we use for each aggregated time point t the parameter value of the corresponding original time point $\vartheta(t)$ in Φ and thereby complete the aggregated instance Φ^{agg} .

Using the SMIP Algorithm 2, we determine a set of valid binary assignments for the aggregated problem instance Φ^{agg} , where we use the parameter n^{maxItAgg} to specify the maximum number of iterations to use in

Algorithm 4: Sequential aggregated horizon heuristic algorithm (AH)**Data:** An instance Φ of model \mathcal{P} **Parameters:** Size of the aggregated horizon n^{aggHor} Maximum number of iterations for the aggregated horizon MIP n^{maxItAgg} Maximum number of iterations for the full horizon MIP $n^{\text{maxItFull}}$ Absolute convergence tolerance $\varepsilon^{\text{tolAbs}}$ Relative convergence tolerance $\varepsilon^{\text{tolRel}}$ **Result:** A list of valid binary assignments for Φ

- 1 Create the time aggregated instance Φ^{agg} with a horizon of length n^{aggHor} from Φ
- 2 Find aggregated binary assignments **aggregatedBAs** by applying SMIP to Φ^{agg} using n^{maxItAgg} , $\varepsilon^{\text{tolAbs}}$, and $\varepsilon^{\text{tolRel}}$
- 3 Remove duplicates from **aggregatedBAs** w.r.t. the to be fixed binary variables defined as {

$$\begin{aligned} m_a^{\text{op}}, m_a^{\text{ac}}, m_a^{\text{cl}} & \quad \forall a \in \mathcal{A}^{\text{arRg}}, \\ m_a^{\text{op}} & \quad \forall a \in \mathcal{A}^{\text{vaOr}}, \\ x_a^{\text{arc}} & \quad \forall a \in \mathcal{A}^{\text{ar}}, \\ x_c^{\text{cfg}} & \quad \forall c \in \mathcal{C}_a^{\text{ar}} \quad \forall a \in \mathcal{A}^{\text{arCp}}, \\ x_s^{\text{st}} & \quad \forall s \in \mathcal{S} \end{aligned}$$
- // Complete the time aggregated BA to BAs for the entire horizon
- 4 **for** **aggBA** \in **aggregatedBAs** **do**
- 5 Create Φ^{fixed} from Φ by fixing the mentioned variables to their values from **aggBA**
- 6 Find binary assignments for Φ by applying SMIP to Φ^{fixed} using $n^{\text{maxItFull}}$, $\varepsilon^{\text{tolAbs}}$, and $\varepsilon^{\text{tolRel}}$, as well as an initial linearization and partial start solution based on **aggBA**
- 7 **return** all determined binary assignments for Φ

SMIP. The general idea for completing aggregated binary assignments to full binary assignments for the original problem instance Φ is to fix certain variables in the model and thereby create the problem instance Φ^{fixed} . We then determine a solution for this instance to generate the binary values for the rest of the original time horizon. Our set of variables to fix for creating Φ^{fixed} is given in Line 3 of Algorithm 4.

Since we do not fix all the variables, the aggregated binary assignments created from solutions for Φ^{agg} might be identical with respect to the variable fixations, which is why we remove duplicated ones in Line 3 of Algorithm 4. Afterward, we create a variant Φ^{fixed} of the original problem instance Φ for each remaining binary assignment in Line 5. When denoting the binary assignment by B and the value of a binary variable x in B by $\varsigma(x, B)$, we perform the following fixations:

$$\begin{aligned} \forall t^{\text{agg}} \in \{0, 1, \dots, n^{\text{aggHor}}\} \text{ and variables } x \in \text{varTypesToFix} : & \quad x_{\vartheta(t^{\text{agg}})} = \varsigma(x_{t^{\text{agg}}}, B), \\ \forall t^{\text{agg}} \in \{1, \dots, n^{\text{aggHor}}\} \text{ and variables } x \in \text{varTypesToFix} & \\ \text{with } \varsigma(x_{t^{\text{agg}}-1}, B) = \varsigma(x_{t^{\text{agg}}}, B), & \\ \forall t \in \{\vartheta(t^{\text{agg}} - 1) + 1, \dots, \vartheta(t^{\text{agg}}) - 1\} : & \quad x_t = \varsigma(x_{t^{\text{agg}}}, B). \end{aligned}$$

In other words, we apply two types of fixations: First, we fix all binary variables of the corresponding types whose time points are part of the aggregated time horizon to their corresponding solution values defined by B . Second and for all time points in between those time points from the aggregated horizon, we check if a variable has the same solution value with respect to B at the two surrounding time points from the aggregated horizon. If it does, we also fix them to exactly that solution value. Otherwise, we do not fix its value. This means that we allow a variable value change between two aggregated time points to happen at any point in time in the corresponding original time interval.

Finally, we compute feasible solutions for Φ^{fixed} by again using the SMIP Algorithm 2 with the parameter $n^{\text{maxItFull}}$ to specify its maximum number of iterations. However, we make two adjustments to the algorithm. First, we change the initial linearization to still use the constant velocity approximation but determine the coefficient $\alpha_{x,a,t}^q$ for each time t , pipe $a = (\ell, r)$ and endnode $x \in \{\ell, r\}$ based on the pressures and flows of the aggregated solution that created the binary assignment. If $t = \vartheta(t^{\text{agg}})$ for some t^{agg} in the aggregated time horizon, we use the pressures and flows of that aggregated time point. Otherwise, we linearly interpolate the values. As a second

adjustment, we provide a partial starting solution to each attempt to solve a linearized MIP, from which the solver tries to build a complete solution for the model. We use the values for all binary variables x_f^{fd} , which determine the activity of a flow direction f , and m_a^{op} , which represent the mode for all fence node valves a . The values for each entity and time point are created based on the given binary assignment using the same fixation strategy as for the creation of Φ^{fixed} . Since both variables do not have a meaningful value for the initial time point, we use for all original time points, which exist before the first time point with a corresponding aggregated time point, the variable value of that aggregated time point.

The result of Algorithm 4 is the union of binary assignments obtained by the adjusted SMIP in Line 6 for all aggregated binary assignments. Note that the binary assignments obtained by completing different aggregated binary assignments are also different due to the guaranteed variable fixations on those original time points with a corresponding aggregated time point.

3.2.3 Aggregated horizon heuristic using rolling horizon

As the final heuristic based on reduced time horizons, we propose a combination of the previous two algorithms given as Algorithm 5. We refer to it as ARH.

The heuristic is nearly equivalent to the aggregated horizon heuristic of Algorithm 4. As the only difference, the problem instance Φ^{agg} on the aggregated time horizon is tackled by the rolling horizon Algorithm 3 instead of the sequential mixed-integer programming Algorithm 2 in Line 2 of 4. Hence, we are given the parameter n^{rolHor} specifying the length of the rolling horizon.

Algorithm 5: Sequential aggregated and rolling horizon heuristic algorithm (ARH)

Data: An instance Φ of model \mathcal{P}

Parameters: Size of the aggregated horizon n^{aggHor}

Size of the rolling horizon n^{rolHor}

Maximum number of iterations for the aggregated horizon MIP n^{maxItAgg}

Maximum number of iterations for the full horizon MIP $n^{\text{maxItFull}}$

Absolute convergence tolerance $\varepsilon^{\text{tolAbs}}$

Relative convergence tolerance $\varepsilon^{\text{tolRel}}$

Result: A list of valid binary assignments for Φ

- 1 **return** the result of applying AH to Φ using n^{aggHor} , n^{maxItAgg} , $n^{\text{maxItFull}}$, $\varepsilon^{\text{tolAbs}}$, and $\varepsilon^{\text{tolRel}}$ as well as using RH with n^{rolHor} instead of SMIP to determine aggregated binary assignments
-

3.3 A dynamic node limit

While testing the above-given heuristics, we observed that during the solving process of the linearized MIP models, we find good primal solutions fast but then spend much time proving their optimality. This is typical behavior for branch-and-bound-based MIP solvers [2]. However, for our heuristics, we are only interested in a good primal solution for each MIP. The corresponding dual bound is not relevant with respect to our underlying application. Therefore, there is the potential to save time if we end the solving process at the point at which the solver has found a good solution and changed its focus to improving the dual bound.

To implement the described behavior, we define a maximum limit L on the number of processed branch-and-bound nodes, which depends on the so far found feasible solutions and can be applied to the MIP solving processes in the heuristics. The general idea is to increase the node limit as long as we find new best primal solutions that significantly improve the primal bound. If this is no longer the case, we hit the dynamic node limit and stop the run with the current best solution.

At the start of the solving process, we set the limit to $L = \infty$. Whenever we find a new best feasible solution with objective function value O at node N , we update the node limit dynamically as follows: Let O^{last} be the objective function value of the last feasible solution that updated the node limit and let $O^{\text{last}} = \infty$ if no feasible solution has been found so far. Then the update can be stated as

$$\text{If } \frac{O^{\text{last}} - O}{O^{\text{last}}} > \alpha^{\text{improve}} : \quad L = \max(n^{\text{limitMin}}, (1 + \alpha^{\text{limitRel}}) \cdot N), \quad O^{\text{last}} = O.$$

In other words, we update the node limit L whenever there is a significant relative improvement of at least $\alpha^{\text{improve}} \in (0, 1)$ of the primal bound compared to its value at the last node limit update. The new limit has a minimal value of n^{limitMin} and is otherwise set to $(1 + \alpha^{\text{limitRel}})$ times the current number of processed branch-and-bound nodes. The minimal node limit avoids a premature termination of the solver and should cover the early stage of the solving process, in which the relative increments in the number of nodes are still relatively high.

4 Computational experiments

The performance of the presented algorithms is tested by conducting a series of computational experiments. The corresponding set of instances is based on a real-world network with corresponding initial states as well as inflow and pressure patterns at the boundary nodes, which are provided by our project partner Open Grid Europe (OGE) [34]. From this data, we choose 40 especially challenging instances by finding those with the highest amount of demand changes. After giving a description of the corresponding network features and specifying the instance generation process, we denote the setup of the experiments as well as the used parameters. Then, we present the results of the two conducted experiments: First, we compare the results of the general sequential mixed-integer programming (SMIP) Algorithm 2 used in Algorithm 1 with those of a general-purpose global MINLP solver. Here, we focus purely on the solution quality to investigate if the SMIP algorithm is able to yield solutions close to the proven optimum. In a second experiment, we evaluate the general performance of the reduced time horizon heuristics presented in Section 3.2 while using the general SMIP approach as a reference.

Since the instance data as well as the software containing the algorithmic implementation were created as part of the industry project mentioned above, corresponding confidentiality and proprietary clauses unfortunately prohibit the publication of the sources, binaries, or instance data used for conducting the computational experiments.

4.1 Instance generation

The instances for testing our algorithms are based on a section of the real-world network controlled by our project partner OGE, which was also used in [25]. It has been modified by replacing the original network station topology with the artificial model that was manually created by industry experts. Furthermore, the network outside of network stations is aggregated using the methods presented in [29].

From the cooperation with our project partner OGE, we attain a set of historical topologies of the network, initial network states, and network demands in hourly resolution over a time period of 12 months, covering the majority of the year 2020. The time period is nearly consecutive, having only three periods of 2, 6, and 11 missing days due to problems during data creation. The initial state data for the whole network was created in a simulation-like process based on single measured reference values as described for the data used in [21]. During time period of 12 months, the network topology, the manually created station model, and the parameters of the applied network aggregation changed multiple times. Hence, we are given a slightly different network topology for each instance. An overview of the network characteristics is given in Table 1, in which the minimal and maximal amounts for each element type are given. Most noteworthy, the number of network stations changed from 7 to 10, as three regulator arrangements in the network were replaced by small stations. We note that compared to the instances used in [25], a less aggressive network aggregation was used, resulting in more elements outside of network stations and harder-to-solve instances as a consequence.

In Table 2, we list parameters regarding the single network stations. If at least one of these values changed over time, we give two sets of values for a station representing the corresponding minimal and maximal amounts per quantity. The network stations A to G are the same as in [25], while the newly introduced stations replacing the regulators are named H, I, and J.

■ **Table 1** Network topology statistics. Since the network and manually created station model change over time, we give for each quantity the corresponding minimal and maximal amounts.

	$ \mathcal{V} $	$ \mathcal{V}^+ $	$ \mathcal{V}^- $	$ \mathcal{A} $	$ \mathcal{A}^{\text{pi}} $	$ \mathcal{A}^{\text{vaOr}} $	$ \mathcal{A}^{\text{rg}} $	$ \mathcal{A}^{\text{arSc}} $	$ \mathcal{A}^{\text{arRg}} $	$ \mathcal{A}^{\text{arBiRg}} $	$ \mathcal{A}^{\text{arCp}} $	$ \mathcal{I} $
Min	375	12	114	421	303	0	11	29	20	2	16	7
Max	477	13	162	531	391	1	13	42	28	3	19	10

■ **Table 2** Network station statistics. Since the network and manually created station model may change over time, we give for each station and each quantity the corresponding minimal and maximal counts. If there are no changes in a station, we only give one set of values.

$i \in \mathcal{I}$	$ \mathcal{V}_i^{\text{fn}} $	$ \mathcal{A}_i^{\text{arSc}} $	$ \mathcal{A}_i^{\text{arRg}} $	$ \mathcal{A}_i^{\text{arBiRg}} $	$ \mathcal{A}_i^{\text{arCp}} $	$ \mathcal{S}_i $	$ \mathcal{F}_i $	$\sum_{a \in \mathcal{A}_i^{\text{arCp}}} \mathcal{C}_a^{\text{ar}} $	$ \mathcal{Z}_i $
A	2	1	0	0	2	5	3	7	0
B _{min}	2	1	0	0	3	5	2	4	0
B _{max}	2	1	0	0	3	6	3	4	0
C _{min}	4	2	6	0	1	3	4	1	0
C _{max}	6	4	6	0	1	9	6	1	0
D _{min}	3	2	1	0	2	5	7	2	0
D _{max}	3	6	2	0	3	7	8	3	0
E _{min}	5	1	5	0	1	9	11	2	0
E _{max}	6	8	6	1	2	19	13	3	0
F _{min}	6	6	1	2	2	7	3	6	0
F _{max}	6	6	1	2	3	11	6	12	6
G _{min}	10	15	6	0	5	25	12	18	17
G _{max}	10	16	10	0	5	33	19	18	17
H	2	0	1	0	0	1	1	0	0
I	3	1	1	0	0	1	1	0	0
J	2	1	2	0	0	3	2	0	0

■ **Table 3** Artificial compressor model sizes of the base and our new model based on the used instance set. Numbers vary over time. The corresponding minimal and maximal numbers originate from the same point in the overall time period.

	# Continuous variables	# Binary variables	# Constraints
base _{min}	56	61	236
base _{max}	68	72	278
new _{min}	14	54	151
new _{max}	17	65	184

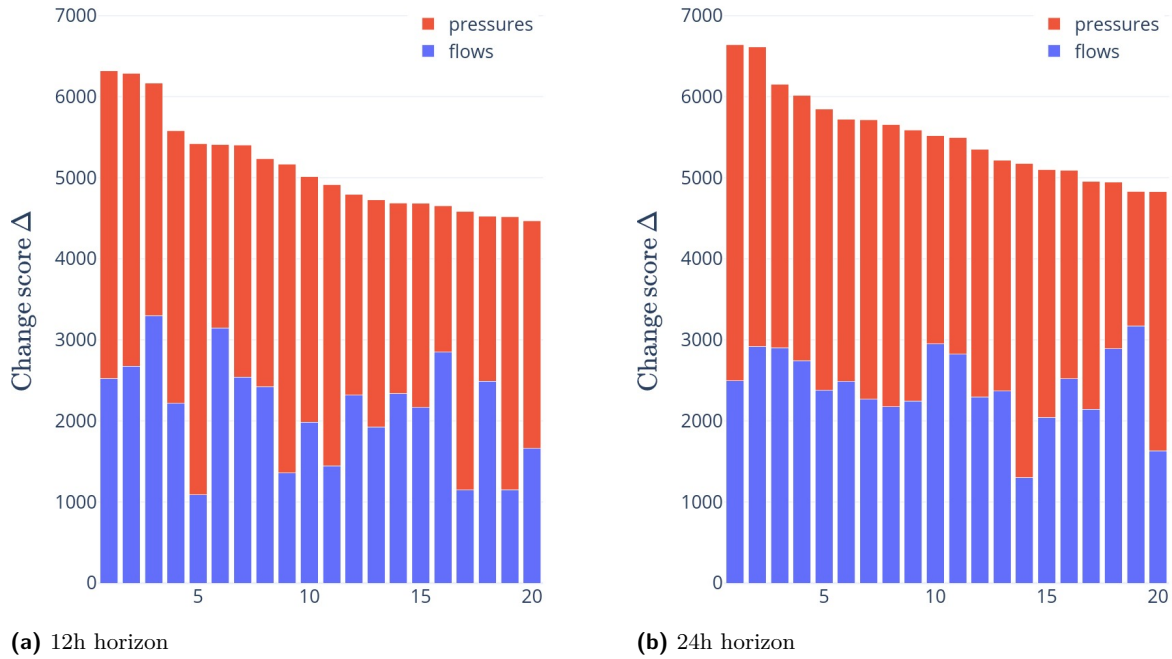
4.1.1 Artificial compressor model size

Based on the specific network station characteristics in the given time period, we compare the actual sizes of the artificial compressors arc models used in the base model of [25] and our newly introduced model from Section 2.5. We count the number of continuous and binary variables as well as the number of linear inequality constraints needed for each discrete time point. For the variables, we included the activation and flow variable per arc as well as all additional variables needed specifically for artificial compressors. The constraints we count for our model are those defined in Equations (11), (12), and (13), while we count all the constraints (31)-(42) from [25] for the base model. Since we count inequality constraints, equations are counted twice. The result is given in Table 3, where we again specified the minimum and maximum numbers over the whole time period. We see that our model is smaller regarding all three categories, as it uses a slightly smaller number of binary variables and considerably fewer continuous variables and inequality constraints.

4.1.2 Challenging demand scenarios

To properly examine if the presented algorithms are suitable for application in time-critical industry environments, we search for particularly challenging instances. In [26], where the gas flow model of [25] was used to conduct a case study on hydrogen transport, the authors suggest that the difficulty of the problem corresponds to the number of element control changes needed to satisfy the given demand scenario. As the number of changes is not known a priori, we instead look for demand scenarios featuring large demand changes for single boundary nodes since we expect these changes to induce inevitable changes in some network elements' control.

To find demand scenarios with large demand changes from within the above-described 12 month time period, we first specify the desired time horizon length for our instances to be 12 and 24 hours. For both, we then



■ **Figure 4** Change scores of the 20 non-overlapping demand scenarios with the highest change score Δ for each of the two lengths of the time horizon. Each bar represents one selected instance and is divided according to the pressure and flow share of Δ , where the pressure is scaled by 100. The instances are sorted by change score. Figures created with [38].

enumerate all time horizons of the corresponding length and sum the differences between the initial and end time point of the two quantities used as future demands in Section 2.8: the boundary node inflow values, which are given as normal volumetric flow Q^0 in $1000\text{m}^3/\text{h}$, and the pressure in bar for entries with positive inflow. The normal volumetric flow is defined as $Q^0 = q/\rho^0$, where ρ^0 is the normal density depending on the gas composition. We then define the final score Δ for the amount of changes of a time horizon by adding the two sums, using the quantities' values for the given units, and weighting the pressure changes by a factor of 100. Finally, we choose from each of the two lists of time horizons a set of 20 demand scenarios by iteratively selecting from the remaining time horizons the one with the highest Δ value, which does not overlap with any of the already selected ones.

An overview of the Δ values for each of the 20 demands scenarios for each time horizon is given in Figure 4. The largest found score value is about 6600, which would be equivalent to a total amount of inflow changes of $6600\ 1000\text{m}^3/\text{h}$ or a total amount of pressure changes of 66 bar at entry nodes over the whole network. The 40 demand scenarios, together with their corresponding network topology and initial network state at the start of the time horizon, define the 40 instances we use as our test set for the computational experiments. When using a time horizon whose time steps are larger than 1 hour, we aggregate the hourly demands as in the function `createTimeAggregatedInstance` of Algorithm 4.

4.2 Setup and parameters

In this section, we give an overview of the computational hardware setup and solver choice, the applied model adjustments to improve its numerical properties, and specify the parameters used for the computational experiments.

4.2.1 Computational setup

The computations were executed on a cluster using 4 cores and 40 GB of RAM of a machine composed of two *Intel Xeon Gold 5122* running at 3.60 GHz. The MIP problems are solved by GUROBI in version 9.5.1 [18], for which we change the general solution strategy to focus on finding primal solutions by using the `MIPFocus` parameter. As the NLP solver, we used IPOPT in version 3.14.3 [42]. Finally, we used SCIP in version 8.0.0 [3] as the global MINLP solver. For SCIP, we apply a constraint feasibility tolerance `numerics/feastol` of 10^{-3} ,

which is even larger than the corresponding default value of 10^{-4} for IPOPT, to improve the acceptance rate for start solutions. We accessed both IPOPT and SCIP via GAMS in version 38.1.0 [14].

4.2.2 Coefficient scaling and rounding

To improve the numeric properties of the models, we use similar ranges for the continuous variables by scaling them such that pressure variables are used in bar and flow values in kg/s. Furthermore, we apply the following modification to the pipes' continuity equation (2a), their momentum equation (2b) in the original or the linearized variant, and the linearized maximum power constraint of artificial configurations (12c): First, we set all coefficients to zero, whose absolute value divided by the largest absolute value coefficient is smaller than 10^{-7} . Afterward, we scale all coefficients of the corresponding constraint such that the absolute smallest non-zero coefficient has an absolute value of 1.

4.2.3 Parameters

For the model, we specify the maximum slack parameters to be 2 bar for the pressure slack. In the case of the inflow, we restrict the possible maximal deviation from the demands to 50%. Hence, we use, instead of a fixed upper bound $\bar{\sigma}^d$ on the inflow slack, an upper bound specific to each boundary node and time point of $\bar{\sigma}_{v,t}^d = 0.5 \cdot |\hat{d}_{v,t}|$ for all boundary nodes $v \in \mathcal{V}^b$ and future time points $t \in \mathcal{T}$. Furthermore, we set the minimum flow value used to determine the absolute velocity in the initial linearization of the friction term to $q^{\min} = 100 \cdot 1000\text{m}^3/\text{h}$.

We define the different weights used in the objective function defined in Equation (31) as

$$\begin{aligned} w^{\sigma-d} &= \frac{100}{\rho^0} \frac{\text{h}}{1000\text{m}^3}, & w^{\sigma-p} &= \frac{1000.0}{\text{bar}}, & w^{\text{ru}} &= w^{\text{arc}} = 50.0, \\ w^{\text{va}} &= w^{\text{rg}} = 500.0, & w^{\text{rg-pl}} &= w^{\text{rg-pr}} = w^{\text{fn-p}} = \frac{10.0}{\text{bar}}, & w^{\text{rg-q}} &= w^{\text{fn-q}} = \frac{1.0}{\rho^0} \frac{\text{h}}{1000\text{m}^3}, \\ w_s^{\text{st}} &\in [0, 4000]. \end{aligned}$$

Note that the multipliers for the flow change and the hourly flow slack are given with respect to normal volumetric flow Q^0 in $1000\text{m}^3/\text{h}$. Hence, we divide them by the normal density ρ^0 to get the corresponding mass flow equivalent. The costs for turning on a simple state of a network station are manually created by the industry experts at OGE, which is why we can only give the range of values for them.

For the heuristic algorithms defined in Section 3, we specify the converge tolerances for the friction term linearization as

$$\varepsilon^{\text{tolRel}} = 0.001, \quad \varepsilon^{\text{tolAbs}} = 0.01 \text{ bar},$$

and the maximum number of sequential mixed-integer programming iterations as

$$n^{\text{maxIt}} = n^{\text{maxItAgg}} = 5, \quad n^{\text{maxItFull}} = 3.$$

The parameters defining the reduced time horizon size are not fixed globally, but define different variants of the heuristics algorithms. Hence, we denote the rolling horizon heuristic of Algorithm 3 with a rolling horizon size of $n^{\text{rolHor}} = X$ by **RXH**, the aggregated horizon heuristic of Algorithm 4 with aggregated horizon size of $n^{\text{aggHor}} = X$ by **AXH**, and the aggregated and rolling horizon heuristic of Algorithm 5 using an aggregated horizon of length $n^{\text{aggHor}} = X$ and a rolling horizon of length $n^{\text{rolHor}} = Y$ by **AXRYH**.

As final parameters, we define the resource limits of the different solving processes, which are given in Table 4. There we distinguish five different model variants: MIP with the dynamic node limit defined in Section 3.3, MIP without this node limit, an NLP run to complete binary assignments, a *fast* NLP variant having a time limit of 5 minutes, and finally, the MINLP run. The given parameters for the dynamic node limit specify that the smallest node limit to stop is 1000 and that a significant primal solution upgrade featuring an objective function value improvement of at least 2% increases the node limit to 150% of the current number of processed nodes. Note that all variants are additionally restricted by the general memory limit of 40 GB.

4.3 Solution quality of general SMIP

In our first experiment, we check the quality of solutions obtained by using the general SMIP Algorithm 2 to determine binary assignments in Algorithm 1 and complete these to a full MINLP solution by an NLP solver.

The lower bound to compare the solutions with is obtained via a general-purpose MINLP solver that gets the SMIP solution as a starting solution. Due to the size of our network and the general complexity of the problem, we were only able to obtain reasonable lower bounds for the smallest time horizon of size 1. Depending on the actual instance, the corresponding MINLP model has the following sizes (rounded to 2 significant digits):

$$\begin{aligned} \# \text{ variables} &\in [2300, 2900], & \# \text{ binary variables} &\in [690, 870], \\ \# \text{ constraints} &\in [3100, 3800], & \# \text{ non-linear constraints} &\in [300, 390]. \end{aligned}$$

Since the run time of the solving process has no priority for this experiment, we use for the single MIP models the parameter variant without the heuristic node limit, and for the NLP the general variant without a time limit, see Table 4.

The result is given in Figure 5. First, we notice that for only 12 out of the 40 instances the MINLP was solved to optimality. In 7 cases, the time limit of roughly one week was hit, while for the remaining 21 instances, the memory limit of 40 GB was exceeded. As a consequence, we do not always have the best possible lower bound available. However, the average gap of 3.2% of the SMIP solutions to that lower bound is still rather small and even closer to the average gap of 2.3% of the MINLP itself. Using the best found primal solutions of the MINLP as a reference for the highest possible lower bound values, the average gap for the SMIP solutions is 0.9%. Regarding run time, Algorithm 1 using the SMIP always finishes in less than 30 seconds, while the geometric mean run time of the MINLP is larger than a day. In summary, the SMIP solutions are very close to the global optimum and can be computed in only a fraction of the global solver's execution time.

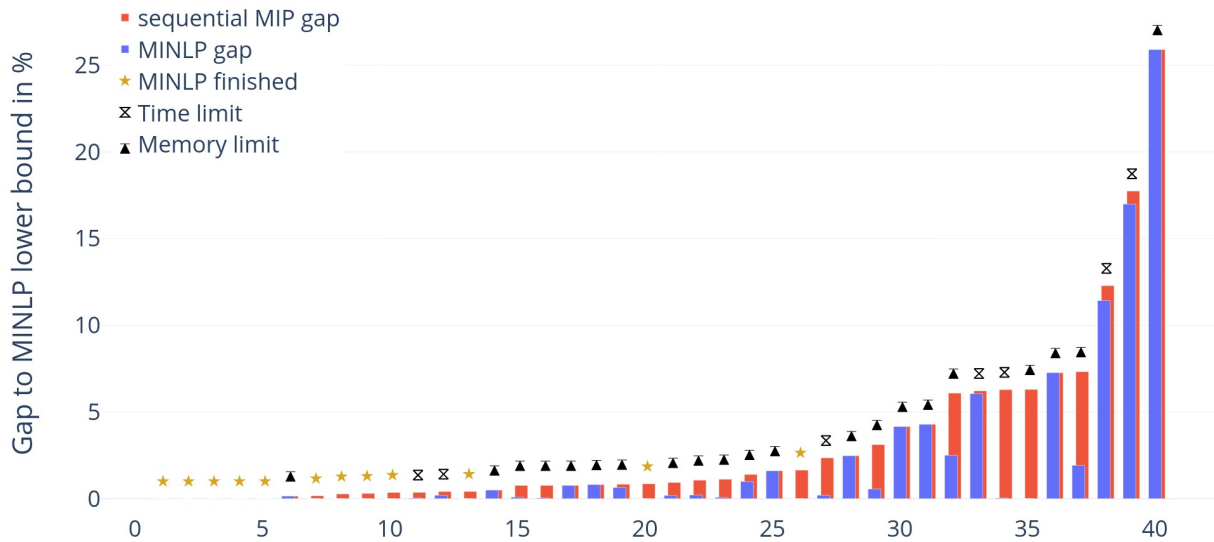


Figure 5 Comparison of the gap of the solutions created by the SMIP Algorithm 2 used in Algorithm 1 with respect to the lower bound found in the MINLP run and the gap of the MINLP itself. Each pair of bars represents one instance, sorted by the gap of the SMIP solution. The values are given in percent. For the instances marked by a star, the MINLP proved optimality. For those instances marked by an hourglass, the MINLP time limit was hit, while for instances marked by a black triangle pointing upwards to a horizontal bar, the MINLP hit the 40 GB memory limit. Figure created with [38].

Table 4 Optimality conditions and resource limits for the different model variants. We distinguish between solving a MIP with and without the dynamic node limit defined in Section 3.3, solving an NLP with and without a time limit, and solving a MINLP.

Variant	Rel. gap	Time limit	n^{limitMin}	α^{limitRel}	α^{improve}
MIP w/node limit	0.0	∞	1000	0.5	0.02
MIP wo/node limit	10^{-2}	86400 s	–	–	–
NLP	–	∞	–	–	–
NLP fast	–	300 s	–	–	–
MINLP	10^{-6}	600000 s	–	–	–

4.4 Performance of reduced time horizon heuristics

In our main experiment of the paper, we evaluate the performance of our proposed algorithm of using a reduced-time-horizon heuristic of Section 3.2 for creating binary assignments to be completed in Algorithm 1. As a reference approach, we use the SMIP for the binary assignment creation, which was shown to find close-to-optimal solutions in the previous section. Note that not having to compare with a global MINLP solver enables us to tackle instances with a time horizon consisting of 12 equally large time steps of 1-hour length for the 12-hour horizon and 2-hour length for the 24-hour horizon. Depending on the actual instance, the underlying MINLP model has the following sizes (rounded to 2 significant digits):

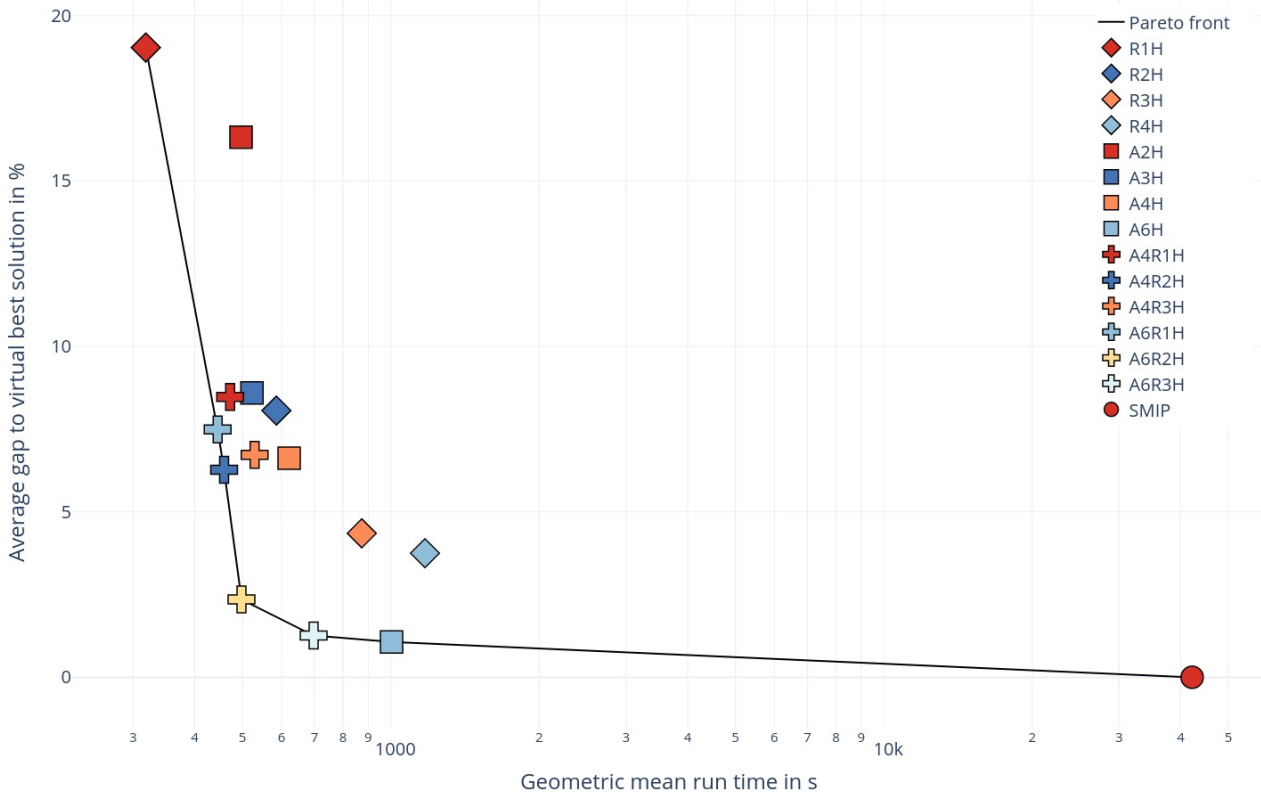
$$\begin{array}{ll} \# \text{ variables} \in [25000, 32000], & \# \text{ binary variables} \in [5600, 7000], \\ \# \text{ constraints} \in [35000, 44000], & \# \text{ non-linear constraints} \in [3600, 4700]. \end{array}$$

After conducting some preliminary experiments, we determined for each of the reduced horizon heuristics the smallest time horizon lengths, which still give reasonably good results, and the largest time horizon lengths, at which the run times are still reasonably short. As a result, we choose the following variants to test for each of the heuristics: The rolling horizon heuristics from R1H to R4H, the aggregated time horizon heuristics A2H, A3H, A4H, and A6H, and for the last two aggregated horizon lengths also the heuristic using the rolling horizon heuristic for some of the submodels with horizon lengths of 1 to 3, resulting in the variants A4R1H, A4R2H, A4R3H, A6R1H, A6R2H, and A6R3H. For all variants, we solve the MIP models using the dynamic branch-and-bound node limit setting and solve the NLP using a time limit, see Table 4. For the reference SMIP approach, we ensure the highest possible solution quality by not using the node limit for MIP or the time limit for NLP.

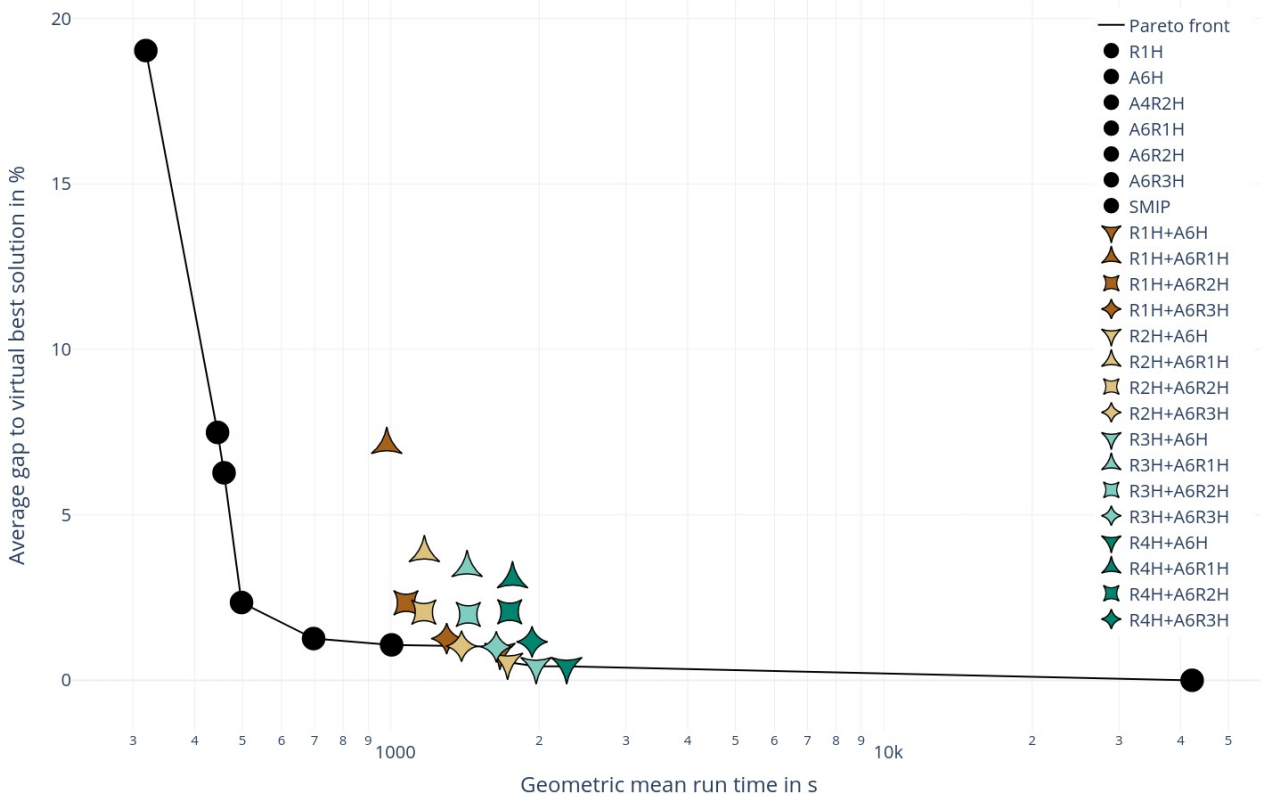
We present the results averaged over all instances for each heuristic variant in two pairs of figures. Each figure shows the position of each variant with respect to the run time and the average gap to the virtual best solution. In the Figures 6, the run time is determined as the geometric mean run time of all the instances and in the Figures 7, we use the maximum run time. If a heuristic failed to find a solution for a specific instance, we used a gap value of 100%. In Table 5, we gave an overview of the number of fails as well as the average results for the single MIP solves and NLP solves per heuristic variant. In the figures, we connected those heuristics being Pareto optimal by a black line, where Pareto optimal heuristics are those not having another heuristic with a smaller average gap and, at the same time, a shorter run time value.

In addition to the individual heuristics displayed in the upper Figures 6a and 7a, we also present in the lower Figures 6b and 7b results for the *combination* of two different heuristic variants. For each combination and instance, we use the best solution found in any of the two heuristics and add up the overall run time of both approaches. As a reference, we also included all Pareto optimal individual heuristics from the corresponding upper pictures marked by black circles. For the combination, we always choose a pair of a pure rolling horizon heuristic and a heuristic using an aggregated time horizon, since we expect the two fundamentally different approaches to have a higher probability of complementing each other in finding high-quality solutions. For the combination, we chose all four rolling horizon heuristics. For the time-aggregation-based heuristics, we used those that are Pareto optimal for the geometric mean and the maximum run time figures, i.e., A6H, A6R1H, A6R2H, and A6R3H. Note that the combined heuristics in the figures have the same color for using the same rolling horizon heuristics and the same symbol for using the same time-aggregation-based heuristic.

The figures show that the reference SMIP approach is the overall best approach in terms of solution quality with an average gap of 0.0%, as it yields the best solution for 39 of the 40 instances and found a solution with a gap of 0.006% for the one not optimal instance. However, it is much slower than every of the reduced time horizon heuristics, both on average and regarding the maximum run time. This is mainly due to the fact that, on average, 1.5 out of the at most 5 solved MIP models reached the 1-day time limit before proving optimality. For the reduced time horizon heuristics, the solution quality is, in general, very good: The average gap to the virtual best solution is below 20% for all considered variants and even below 10% for all but two. The best average gap apart from the general SMIP approach was achieved by the A6H heuristic with a gap of 1.07% and 17/40 instances, for which the best solution has been found. Two variants with similar results are A6R2H and A6R3H, which are close in terms of the average gap but have better run time values. Hence, the summary for the individual variants is to choose an aggregated horizon heuristic with a long time horizon and either use it directly for the best results or internally use the rolling horizon approach for shorter run times at the cost of a slightly worse solution quality. When comparing the geometric mean and the maximum run time, the pictures are very similar, with only two heuristic variants being Pareto optimal for only one of the two. The maximum run time values are between 2 to 5 times higher for all the heuristics except A2H, where the factor is over 7.

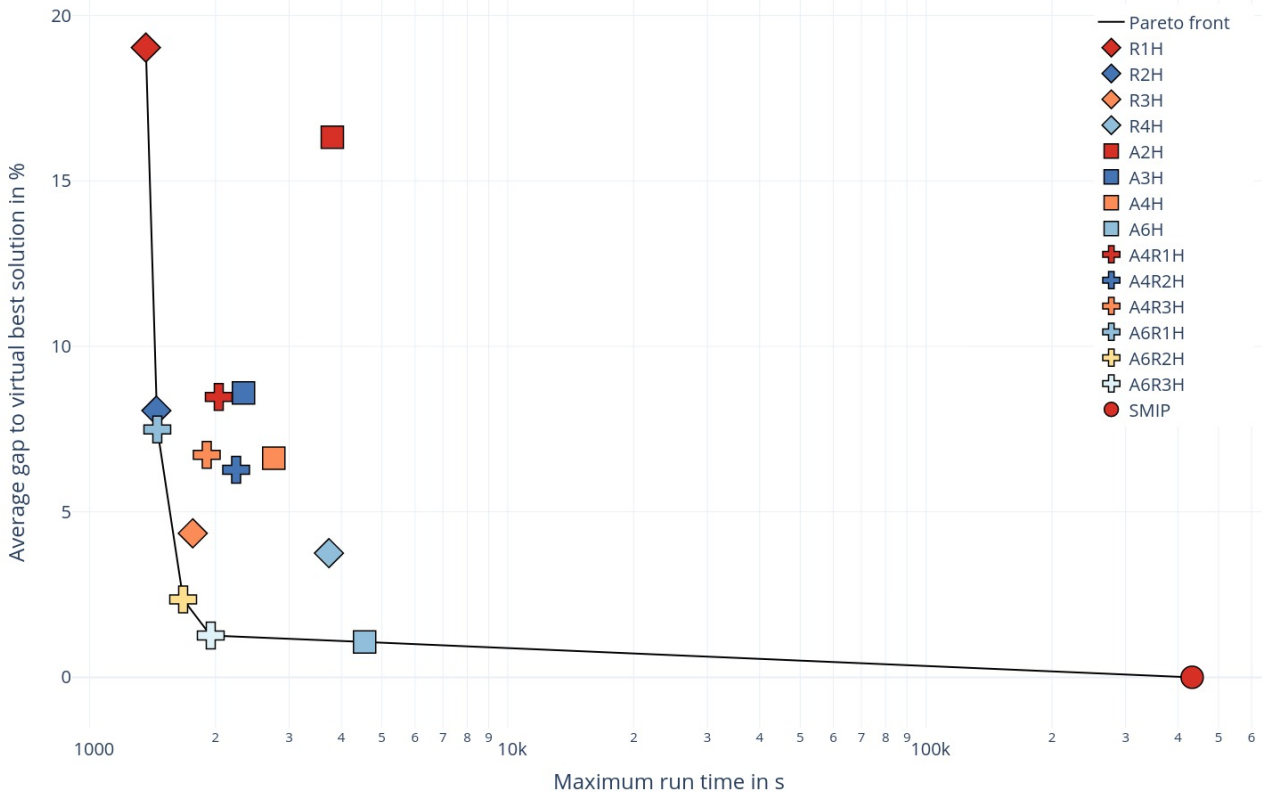


(a) Individual heuristics

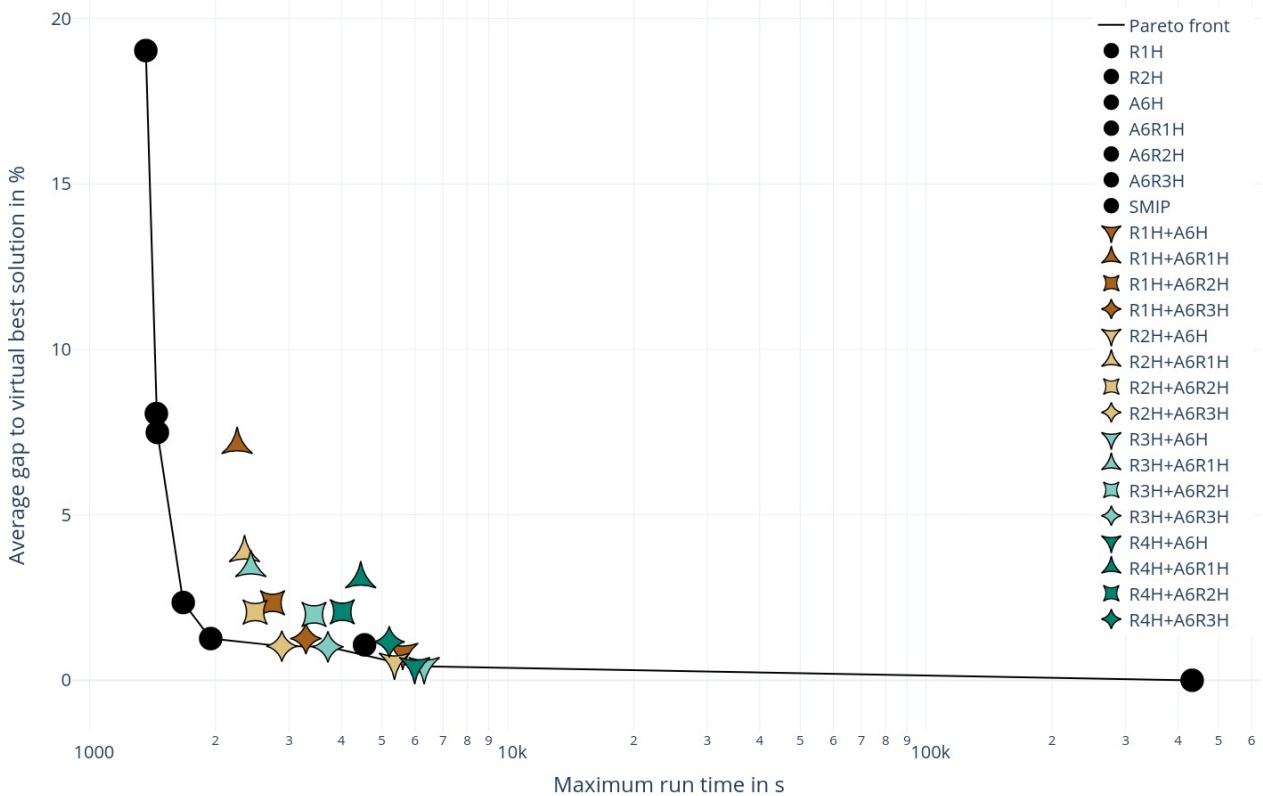


(b) Combine heuristics and Pareto optimal points of the individual heuristics

Figure 6 Position of the heuristic variants in a geometric-mean-run-time vs. gap-to-virtual-best-solution plot. If a heuristic does not find a solution for some instance, we count the corresponding gap as 100%. The Pareto optimal points are linked by a black line. Figures created with [38].



(a) Individual heuristics



(b) Combine heuristics and Pareto optimal points of the individual heuristics

Figure 7 Position of the heuristic variants in a maximum-run-time vs. gap-to-virtual-best-solution plot. If a heuristic does not find a solution for some instance, we count the corresponding gap as 100%. The Pareto optimal points are linked by a black line. Figures created with [38].

■ **Table 5** Statistics for the heuristic variants. The first two columns state the heuristic’s name and the number of instances for which it fails to find a feasible solution. The next four columns state the average number of MIP models solved per instance, followed by the corresponding average number of those MIPs finished with an optimal solution, finished due to reaching the dynamic node limit, or finished with proven infeasibility of the corresponding model. Next, we have a similar block of four columns for the average number of NLP models solved per instance. We state the total average (which is equal to the average number of binary assignments found), the average number of converged NLPs, the average number of NLP finishing with a feasible solution at the given time limit, and the average number of failed attempts to find a feasible solution within the time limit. The final two columns depict the geometric mean run time for the NLP in seconds, first for a single NLP run and second for the summed run time of all NLP runs per instance. If a heuristic did not produce any binary assignments, the NLP run time was set to 1 second.

	Fail	# Avg MIP runs				# Avg NLP runs				Mean NLP time	
		Total	Opt	DNL	Inf	Total	Conv	Time	Fail	by NLP	by inst
R1H	4	46.73	46.62	0.00	0.10	3.38	3.12	0.05	0.20	161.6	307.4
R2H	1	46.08	45.35	0.70	0.03	3.67	3.58	0.03	0.07	132.8	418.6
R3H	0	44.00	41.42	2.58	0.00	3.88	3.75	0.05	0.07	130.3	474.9
R4H	0	38.25	30.80	7.45	0.00	3.77	3.70	0.03	0.05	120.8	430.9
A2H	1	10.40	8.82	1.48	0.10	4.55	4.45	0.05	0.05	86.6	258.3
A3H	1	11.22	8.82	2.33	0.07	4.65	4.65	0.00	0.00	78.3	246.6
A4H	0	11.68	8.90	2.67	0.10	4.55	4.55	0.00	0.00	77.5	278.0
A6H	0	12.62	9.15	3.42	0.05	4.90	4.85	0.00	0.05	76.6	315.6
A4R1H	0	24.55	24.48	0.07	0.00	4.78	4.78	0.00	0.00	71.4	278.6
A4R2H	0	19.02	18.62	0.40	0.00	4.47	4.42	0.00	0.05	75.5	245.8
A4R3H	0	15.03	12.57	2.42	0.03	4.47	4.45	0.03	0.00	75.9	260.5
A6R1H	1	33.77	33.75	0.00	0.03	5.35	5.33	0.00	0.03	74.4	285.5
A6R2H	0	28.12	27.38	0.72	0.03	4.75	4.75	0.00	0.00	70.3	266.8
A6R3H	0	25.23	22.52	2.70	0.00	5.08	5.05	0.00	0.03	76.9	324.8

For the two variants A6R2H and A6R3H with the lowest run times of those with an average gap below 3%, the maximum run time is about half an hour, which is very fast for these types of problems.

Using two heuristics in combination improves the overall solution quality as expected, with 6 of the 16 combinations having an average gap better than the best individual solution and all of the combinations involving A6H reaching an average gap below 1%. However, this comes at the cost of rather large run time increases. Only the fastest of the combined heuristics has a better geometric run time than the one of the slowest Pareto optimal reduced time horizon heuristic. However, in that larger run time segment, several Pareto optimal combinations of the A6R3H and A6H heuristic combined with the rolling horizon heuristics can be found for both run time metrics. The best of them decrease the gap to the SMIP variant to a minimum of 0.42% for the two combinations of R3H and R4H with A6H while still being much faster than SMIP itself.

The reason for the overall bad run time for the combination of two heuristics is the rather long solve time for the NLP model, as displayed in Table 5. For heuristics like R1H and R2H that determine binary assignments very fast, the geometric mean NLP time per instance is equal to 96.5% and 71.5% of the geometric mean overall run time, respectively. Hence, testing more binary assignments for yielding good overall solutions to the problem is apparently a considerable time investment.

To compare our algorithm with a procedure similar to the one proposed in [25], we also tested the SMIP approach with only a single iteration, i.e., $n^{\max It} = 1$. This is close to the algorithm proposed there, as the linearization used for the model determining the binary solution values is only updated in the rare case of failure in the subsequent procedure searching for a non-linear solution for the found binary assignment. The SMIP heuristic with $n^{\max It} = 1$ has an average gap of 42.89% to the virtual best solution and is, therefore, worse than all of our proposed heuristic variants. For 14 out of the overall 40 instances, it reached the time limit of 1 day. This leads to a geometric mean run time of 9541.0 seconds, which is again considerably larger than those of all our heuristics variants. Hence, we can conclude that our proposed algorithms are clearly superior.

5 Outlook

In this article, we presented several improvements for finding high-quality solutions for the control optimization problem on transient gas networks in a time-critical environment. First, we introduced a new model for the artificial arcs representing the compressors, which is configuration-based and closer to the original compression capabilities while using fewer variables and constraints on real-world instances. Furthermore, we propose a new heuristic algorithm for the problem. It first determines a set of solution values for all binary variables, which we call a binary assignment, and afterward solves a corresponding NLP to complete it to a full solution of the problem. The binary assignments are obtained using sequential mixed-integer programming, two heuristics based on reduced time horizons, and a specialized dynamic node limit. Our approach was tested on particularly challenging real-world instances featuring a large amount of supply and demand changes. The results confirm that we are able to find solutions close to the global optimum while being very fast compared to general-purpose MINLP solvers and a comparable approach from the literature.

We see two main reasons for the success of our heuristic approach for the given problem. First, we recognized that from the multitude of possible future control recommendations, only a few combinations fit a given scenario. This is especially true for the given objective function that mainly punishes changes of control modes and the state of the network at specific nodes. The reason for this is the limited gas velocity, which causes the consequences of control changes to have a delayed realization. Second, the network's pipes act as gas storage and can therefore compensate for a local shortage or surplus of gas for a short amount of time. For this reason, changing the control of an element a little too early or too late is often feasible and only causes a small objective function value penalty. This fact is beneficial for both heuristic approaches: The rolling horizon heuristic results in too late control changes due to not having a full view of the future. In the case of the aggregated horizon, the time points in which control decisions can be made are limited. Hence, they may be a little late or a little early. Due to the pipe storage, solutions with these control time offsets are likely to be similar to one of the few reasonable control recommendations while only having a slightly worse objective function value.

Future research on the overall topic can continue in many directions. Model-wise, we could further refine the compressor model to guarantee that the corresponding feasible region is an actual relaxation of the original feasible region. Another possibility to improve the accuracy would be to also update the linearization of the power bound during the SMIP and include the original non-linear formula in the final NLP model. Similarly, a more complex and hyperbolic pipe model like the one used in [17] could be used. However, as we have seen in the results, the run time for solving the NLP models is already quite high. Therefore, it might be beneficial to just complete the most promising binary assignments to full solutions, for example, by comparing their corresponding linearized MIP solution value, or to only complete one of a set of similar binary assignments. Another possibility would be to use specialized approaches like the one described in [10] for solving the NLP problem instead of general-purpose solvers. Regarding the instance set used in our computational experiments, it would be interesting to examine if heterogeneous step lengths in the time horizon lead to more complex solutions or to evaluate our algorithm in the context of pure hydrogen networks similar to those presented in [26]. Finally, we mention the inclusion of the uncertainty of future demands as a possible extension of the examined problem, which was for the stationary case investigated in [1].

A Appendix

Table 6 List of all used variables, specifying their domain, meaning and unit. Note that 1 bar = 10^5 Pa.

Variable	Meaning	Unit
$p_{v,t} \in \mathbb{R}_{\geq 0}$	Pressure at node $v \in \mathcal{V}$	bar
$q_{v,a,t} \in \mathbb{R}$	Flow into or out of pipe $a = (l, r) \in \mathcal{A}^{\text{pi}}$ at endnode $v \in \{l, r\}$	kg/s
$q_{a,t} \in \mathbb{R}$	Flow over arc $a \in \mathcal{A} \setminus \mathcal{A}^{\text{pi}}$	kg/s
$d_{v,t} \in \mathbb{R}$	Inflow into boundary node $v \in \mathcal{V}^{\text{b}}$	kg/s
$m_{a,t}^{\text{op}} \in \{0, 1\}$	Selection of open mode for $a \in \mathcal{A}^{\text{va}} \cup \mathcal{A}^{\text{rg}}$	1
$m_{a,t}^{\text{cl}} \in \{0, 1\}$	Selection of closed mode for $a \in \mathcal{A}^{\text{rg}}$	1
$m_{a,t}^{\text{ac}} \in \{0, 1\}$	Selection of active mode for $a \in \mathcal{A}^{\text{rg}}$	1
$x_{a,t}^{\text{arc}} \in \{0, 1\}$	Activity of artificial arc $a \in \mathcal{A}^{\text{ar}}$	1
$x_{a,t}^{\text{fwd}} \in \{0, 1\}$	Selection of forward direction for $a \in \mathcal{A}^{\text{arBiRg}}$	1
$x_{a,t}^{\text{bwd}} \in \{0, 1\}$	Selection of backward direction for $a \in \mathcal{A}^{\text{arBiRg}}$	1
$q_{a,t}^{\text{fwd}} \in \mathbb{R}_{\geq 0}$	Forward flow of $a \in \mathcal{A}^{\text{arBiRg}}$	kg/s
$q_{a,t}^{\text{bwd}} \in \mathbb{R}_{\geq 0}$	Backward flow of $a \in \mathcal{A}^{\text{arBiRg}}$	kg/s
$x_{c,t}^{\text{cfg}} \in \{0, 1\}$	Selection of configuration $c \in \mathcal{C}_a^{\text{ar}}$ for $a \in \mathcal{A}^{\text{arCp}}$	1
$x_{s,t}^{\text{st}} \in \{0, 1\}$	Selection of simple state $s \in \mathcal{S}$	1
$x_{f,t}^{\text{fd}} \in \{0, 1\}$	Selection of flow direction $f \in \mathcal{F}$	1
$\sigma_{v,t}^{p+} \in \mathbb{R}_{\geq 0}$	Positive pressure slack for boundary node $v \in \mathcal{V}^{\text{b}}$	bar
$\sigma_{v,t}^{p-} \in \mathbb{R}_{\geq 0}$	Negative pressure slack for boundary node $v \in \mathcal{V}^{\text{b}}$	bar
$\sigma_{v,t}^{d+} \in \mathbb{R}_{\geq 0}$	Positive inflow slack for boundary node $v \in \mathcal{V}^{\text{b}}$	kg/s
$\sigma_{v,t}^{d-} \in \mathbb{R}_{\geq 0}$	Negative inflow slack for boundary node $v \in \mathcal{V}^{\text{b}}$	kg/s
$\delta_{a,t}^{\text{va}} \in \{0, 1\}$	Mode change for original valve $a \in \mathcal{A}^{\text{vaOr}}$	1
$\delta_{a,t}^{\text{rg}} \in \{0, 1\}$	Mode change for regulator $a \in \mathcal{A}^{\text{rg}}$	1
$\delta_{a,t}^{\text{rg-pl}} \in \mathbb{R}_{\geq 0}$	Change of incoming pressure of active $a \in \mathcal{A}^{\text{rg}}$	bar
$\delta_{a,t}^{\text{rg-pi}} \in \mathbb{R}_{\geq 0}$	Change of outgoing pressure of active $a \in \mathcal{A}^{\text{rg}}$	bar
$\delta_{a,t}^{\text{rg-qa}} \in \mathbb{R}_{\geq 0}$	Change of flow over of active $a \in \mathcal{A}^{\text{rg}}$	kg/s
$\delta_{a,t}^{\text{arc}} \in \{0, 1\}$	Activity change for artificial arc $a \in \mathcal{A}^{\text{ar}}$	1
$\delta_{s,t}^{\text{st}} \in \{0, 1\}$	Activation of simple state $s \in \mathcal{S}$	1
$\delta_{a,t}^{\text{fn-p}} \in \mathbb{R}_{\geq 0}$	Change of pressure of fence node $v \in \mathcal{V}_i^{\text{fn}}$ for $i \in \mathcal{I}$	bar
$\delta_{a,t}^{\text{fn-q}} \in \mathbb{R}_{\geq 0}$	Change of flow over valve of fence node $v \in \mathcal{V}_i^{\text{fn}}$ for $i \in \mathcal{I}$	kg/s

References

- 1 Dennis Adelhütte, Denis Aßmann, Tatiana González Grandón, Martin Gugat, Holger Heitsch, René Henrion, Frauke Liers, Sabrina Nitsche, Rüdiger Schultz, Michael Stingl, and David Wintergerst. Joint Model of Probabilistic-Robust (Proburst) Constraints Applied to Gas Network Optimization. *Vietnam J. Math.*, 49(4):1097–1130, 2021.
- 2 Timo Berthold, Gregor Hendel, and Thorsten Koch. From Feasibility to Improvement to Proof: Three Phases of Solving Mixed-Integer Programs. *Optim. Methods Softw.*, 33(3):499–517, 2018.
- 3 Ksenia Bestuzheva, Mathieu Besançon, Wei-Kun Chen, Antonia Chmiela, Tim Donkiewicz, Jasper van Doornmalen, Leon Eifler, Oliver Gaul, Gerald Gamrath, Ambros Gleixner, Leona Gottwald, Christoph Graczyk, Katrin Halbig, Alexander Hoen, Christopher Hojny, Rolf van der Hulst, Thorsten Koch, Marco Lübbecke, Stephen J. Maher, Frederic Matter, Erik Mühmer, Benjamin Müller, Marc E. Pfetsch, Daniel Rehfeldt, Steffan Schlein, Franziska Schläpfer, Felipe Serrano, Yuji Shinano, Boro Sofranac, Mark Turner, Stefan Vigerske, Fabian Wegscheider, Philipp Wellner, Dieter Weninger, and Jakob Witzig. The SCIP Optimization Suite 8.0. ZIB-Report 21-41, ZIB, 2021.
- 4 Natachia Boland, Andreas Ernst, T. Kalinowski, Mateus Rocha de Paula, Martin Savelsbergh, and Gaurav Singh. Time Aggregation for Network Design to Meet Time-Constrained Demand. In *MODSIM 2013: 20th International Congress on Modelling and Simulation - Adapting to Change: The Multiple Roles of Modelling*, pages 3281–3287. Modelling and Simulation Society of Australia and New Zealand, 2013.

- 5 Pierre Bonami, Andrea Lodi, Andrea Tramontani, and Sven Wiese. On Mathematical Programming with Indicator Constraints. *Math. Program.*, 151(1):191–223, 2015.
- 6 Robert Burlacu, Herbert Egger, Martin Groß, Alexander Martin, Marc E. Pfetsch, Lars Schewe, Mathias Sirvent, and Martin Skutella. Maximizing the Storage Capacity of Gas Networks: A Global MINLP Approach. *Optim. Eng.*, 20(2):543–573, 2019.
- 7 Florin Capitanescu. A Relax and Reduce Sequential Decomposition Rolling Horizon Algorithm to Value Dynamic Network Reconfiguration in Smart Distribution Grid. In *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, pages 1–6, 2017.
- 8 Pia Domschke, Björn Geißler, Oliver Kolb, Jens Lang, Alexander Martin, and Antonio Morsi. Combination of Nonlinear and Linear Optimization of Transient Gas Networks. *INFORMS J. Comput.*, 23(4):605–617, 2011.
- 9 Pia Domschke, Benjamin Hiller, Jens Lang, Volker Mehrmann, Riccardo Morandin, and Caren Tischendorf. Gas Network Modeling: An Overview. Technical Report, Technische Universität Darmstadt, 2021.
- 10 Pia Domschke, Oliver Kolb, and Jens Lang. Fast and Reliable Transient Simulation and Continuous Optimization of Large-Scale Gas Networks. *Math. Methods Oper. Res.*, 95(3):475–501, 2022.
- 11 Federal Ministry for Economic Affairs and Climate Action. KfW, Gasunie and RWE Sign MoU to Build an LNG Terminal at Brunsbüttel. <https://www.bmwi.de/Redaktion/EN/Pressemitteilungen/2022/03/202203-kfw-gasunie-and-rwe-sign-mou-to-build-an-lng-terminal-at-brunsbuettel.html>, 2022. Accessed: 2022-03-24.
- 12 FNB Gas – Association of German transmission system operators. Scenario Framework Gas Network Development Plan 2022–2032. <https://fnb-gas.de/en/scenario-framework/scenario-framework-2022/>, 2021. Accessed: 2022-03-24.
- 13 Armin Fügenschuh, Björn Geißler, Ralf Gollmer, Antonio Morsi, Marc E. Pfetsch, Jessica Rövekamp, Martin Schmidt, Klaus Spreckelsen, and Marc C. Steinbach. Physical and Technical Fundamentals of Gas Networks. In Thorsten Koch, Benjamin Hiller, Marc E. Pfetsch, and Lars Schewe, editors, *Evaluating Gas Network Capacities*, volume 21 of *MOS-SIAM Series on Optimization*. Society for Industrial and Applied Mathematics, 2015.
- 14 GAMS Development Corporation. General Algebraic Modeling System (GAMS) Release 38.1.0. Fairfax, VA, USA, 2022.
- 15 Lukas Glomb, Frauke Liers, and Florian Rösel. A Rolling-Horizon Approach for Multi-Period Optimization. *Eur. J. Oper. Res.*, 300(1):189–206, 2022.
- 16 Ángel M. González Rueda, Julio González Díaz, and María P. Fernández de Córdoba. A Twist on SLP Algorithms for NLP and MINLP Problems: An Application to Gas Transmission Networks. *Optim. Eng.*, 20(2):349–395, 2019.
- 17 Martin Gugat, Günter Leugering, Alexander Martin, Martin Schmidt, Mathias Sirvent, and David Wintergerst. MIP-Based Instantaneous Control of Mixed-Integer PDE-Constrained Gas Transport Problems. *Comput. Optim. Appl.*, 70(1):267–294, 2018.
- 18 Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, Version 9.5. <https://www.gurobi.com>, 2021. Accessed: 2022-03-24.
- 19 Mirko Hahn, Sven Leyffer, and Victor M. Zavala. Mixed-Integer PDE-Constrained Optimal Control of Gas Networks. Preprint, Argonne National Laboratory, 2017.
- 20 Felix Hennings. Benefits and Limitations of Simplified Transient Gas Flow Formulations. In *Operations Research Proceedings 2017*, pages 231–237. Springer, 2018.
- 21 Felix Hennings. Large-Scale Empirical Study on the Momentum Equation’s Inertia Term. *J. Nat. Gas Sci. Eng.*, 95: article no. 104153, 2021.
- 22 Felix Hennings, Lovis Anderson, Kai Hoppmann-Baum, Mark Turner, and Thorsten Koch. Controlling Transient Gas Flow in Real-World Pipeline Intersection Areas. *Optim. Eng.*, 22(2):687–734, 2021.
- 23 Felix Hennings, Milena Petkovic, and Tom Streubel. On the Numerical Treatment of Interlaced Target Values - Modeling, Optimization and Simulation of Regulating Valves in Gas Networks. ZIB-Report 21-32, ZIB, 2021.
- 24 Kai Hoppmann-Baum. *Mathematical Programming for Stable Control and Safe Operation of Gas Transport Networks*. PhD thesis, Technische Universität Berlin, 2022.
- 25 Kai Hoppmann-Baum, Felix Hennings, Ralf Lenz, Uwe Gotzes, Nina Heinecke, Klaus Spreckelsen, and Thorsten Koch. Optimal Operation of Transient Gas Transport Networks. *Optim. Eng.*, 22(2):735–781, 2021.
- 26 Kai Hoppmann-Baum, Felix Hennings, Janina Zittel, Uwe Gotzes, Eva-Maria Spreckelsen, Klaus Spreckelsen, and Thorsten Koch. From Natural Gas towards Hydrogen - A Feasibility Study on Current Transport Network Infrastructure and Its Technical Control. ZIB-Report 20-27, ZIB, 2020.
- 27 Thorsten Koch, Benjamin Hiller, Marc E. Pfetsch, and Lars Schewe, editors. *Evaluating Gas Network Capacities*, volume 21 of *MOS-SIAM Series on Optimization*. Society for Industrial and Applied Mathematics, 2015.
- 28 Oliver Kolb, Jens Lang, and Pia Bales. An Implicit Box Scheme for Subsonic Compressible Flow with Dissipative Source Term. *Numer. Algorithms*, 53(2):293–307, 2010.
- 29 Ralf Lenz. *Optimization of Stationary Expansion Planning and Transient Network Control by Mixed-Integer Nonlinear Programming*. PhD thesis, Technische Universität Berlin, 2021.

- 30 Debora Mahlke, Alexander Martin, and Susanne Moritz. A Simulated Annealing Algorithm for Transient Optimization in Gas Networks. *Math. Methods Oper. Res.*, 66(1):99–115, 2007.
- 31 Susanne Moritz. *A Mixed Integer Approach for the Transient Case of Gas Network Optimization*. Doctoral Thesis, Technische Universität Darmstadt, 2007.
- 32 Alexandra M. Newman and Mark Kuchta. Using Aggregation to Optimize Long-Term Production Planning at an Underground Mine. *Eur. J. Oper. Res.*, 176(2):1205–1218, 2007.
- 33 Johann Nikuradse. *Laws of Flow in Rough Pipes*. National Advisory Committee for Aeronautics Washington, 1950.
- 34 Open Grid Europe GmbH. <https://oge.net/en>. Accessed: 2022-03-24.
- 35 Andrej J. Osiadacz. Different Transient Flow Models - Limitations, Advantages, And Disadvantages. In *PSIG Annual Meeting*. Pipeline Simulation Interest Group, 1996. PSIG-9606.
- 36 J. Pápay. A Termeléstechnológiai Paraméterek Változása a Gáztelepek Művelése Során. *OGIL Műsz. Tud. Közl.*, pages 267–273, 1968.
- 37 Marc E. Pfetsch, Armin Fügenschuh, Björn Geißler, Nina Geißler, Ralf Gollmer, Benjamin Hiller, Jesco Humpola, Thorsten Koch, Thomas Lehmann, Alexander Martin, Antonio Morsi, Jessica Rövekamp, Lars Schewe, Martin Schmidt, Rüdiger Schultz, Robert Schwarz, Jonas Schweiger, Claudia Stangl, Marc C. Steinbach, Stefan Vigerske, and Bernhard M. Willert. Validation of Nominations in Gas Network Optimization: Models, Methods, and Solutions. *Optim. Methods Softw.*, 30(1):15–53, 2015.
- 38 Plotly Technologies Inc. Collaborative Data Science. <https://plotly.com>, 2015. Accessed: 2022-03-24.
- 39 Gabor Takacs. Comparing Methods for Calculating Z-factor. *Oil Gas J.*, 87(20):43–46, 1989.
- 40 Till Tantau. The TikZ and PGF Packages - Manual for Version 3.1.9a. <https://github.com/pgf-tikz/pgf>, 2021. Accessed: 2022-03-24.
- 41 Lorenzo Tiacchi and Stefano Saetta. Demand Forecasting, Lot Sizing and Scheduling on a Rolling Horizon Basis. *Int. J. Prod. Econ.*, 140(2):803–814, 2012.
- 42 Andreas Wächter and Lorenz T. Biegler. On the Implementation of an Interior-Point Filter Line-Search Algorithm for Large-Scale Nonlinear Programming. *Math. Program.*, 106(1):25–57, 2006.
- 43 Tom Walther and Benjamin Hiller. Modelling Compressor Stations in Gas Networks. ZIB-Report 17-67, ZIB, 2017.