Math. Meth. Oper. Res. (2007) 66: 99–115
DOI 10.1007/s00186-006-0142-9

ORIGINAL ARTICLE

Debora Mahlke · Alexander Martin
Susanne Moritz

A simulated annealing algorithm for transient optimization in gas networks

Received: 24 February 2006 / Revised: 18 October 2006 /
Published online: 23 January 2007

Abstract In this paper we present a simulated annealing approach for the gas network optimization problem. A gas network consists of a set of pipes to transport the gas from the sources to the sinks whereby gas pressure gets lost due to friction. Further on there are compressors, which increase gas pressure, and valves. The aim is to minimize fuel gas consumption of the compressors whereas demands of consumers have to be satisfied. The problem of transient (time-dependent) optimization of gas networks results in a highly complex mixed integer nonlinear program. We relax the equations describing the gas dynamic in pipes by adding these constraints combined with appropriate penalty factors to the objective function. A suitable neighborhood structure is developed for the relaxed problem where time steps as well as pressure and flow of the gas are decoupled. Our approach convinces with flexibility and very good computational results.

Keywords Mixed integer nonlinear programming · Transient gas optimization · Simulated annealing · Heuristics · Relaxation

1 Introduction

In this paper we consider the following problem. We are given a gas network consisting of compressors and valves that are connected by pipes. There are consumers that need a certain amount of gas at a specified quality and pressure, and sources where some gas is delivered with a certain pressure and volume. While the gas flows through the network, pressure decreases due to friction with the pipe walls.

D. Mahlke · A. Martin (✉) · S. Moritz
Department of Mathematics, TU Darmstadt, Schloßgartenstraße 7,
64289 Darmstadt, Germany
E-mail: mahlke@mathematik.tu-darmstadt.de
E-mail: martin@mathematik.tu-darmstadt.de
E-mail: moritz@mathematik.tu-darmstadt.de

Compressors are used to compensate the pressure loss at the cost of consuming fuel gas (about 2 % of the gas running through them). The aim of the so called transient technical optimization (TTO) is to operate the gas transmission in such a way that the consumer demands are satisfied and the compressors are set in cost-efficiently.

The problem of TTO includes nonlinear and combinatorial aspects. The gas dynamics in pipes and the fuel gas consumption of compressors are highly nonlinear elements. Combinatorial aspects are given by switching modes of compressors and valves.

Most papers that can be found in literature only handle the stationary case where one time step is considered (see for example Borraz-Sánchez and Ríos-Mercado 2005; Jenicek 1993; Martin et al. 2006; Pratt and Wilson 1984; Carter 1998; Smeers and De Wolf 2000; Wright et al. 1998). Optimization of the transient case (several time steps) remains one of the great challenges. Note that from a mathematical point of view in the time-dependent case, even the specification of a feasible solution is a very difficult task. In this paper we present a simulated annealing algorithm for the TTO. The aim of the heuristic is to yield a good feasible solution in adequate runtime.

In Sect. 2 we formulate the problem of TTO as a mixed integer nonlinear problem. In Sect. 3 we introduce the general idea of the simulated annealing algorithm, give a mathematical description of it, and adapt the simulated annealing algorithm to the TTO in Sect. 4. Finally, we conclude with computational results showing the applicability of our developed algorithm in practice.

In the literature various approaches for gas network optimization can be found whereas none covers all nonlinear, combinatorial or time-dependent aspects. As already mentioned most papers concentrate on the stationary case.

In Gopal (1979), Zimmer (1975) and Carter (1998) dynamic programming is used. The first two restrict to a simplified network structure, i. e., a directed graph consisting of pipes and compressors without cycles, and the latter one also comprises series parallel network elements. Because of the nonlinear elements the problem is often tackled by nonlinear optimization methods whereas the combinatorial aspects are neglected. In this context sub-gradient techniques (Jenicek 1993; Králik 1993) are used.

Heuristic approaches can also be found for stationary gas network optimization like for example simulated annealing (Wright et al. 1998) and a hybrid heuristic approach (Borraz-Sánchez and Ríos-Mercado 2005). Wright et al. (1998) concentrates on the optimization of compressor stations, i. e., the authors look for optimal configurations and power settings for a large number of compressors arranged in series or in parallel. Borraz-Sánchez and Ríos-Mercado (2005) combines non-sequential dynamic programming with tabu search in a two-stage iterative procedure. In the first step gas flow variables are fixed and then optimal pressure values are found via non-sequential dynamic programming. Thereafter, in a second stage, pressure values are fixed and a tabu search procedure guides the search in the flow variable space. Pratt and Wilson (1984) applies mixed integer programming for the stationary case. The problem is solved iteratively by means of sequential linear programming whereby the nonlinearities are approximated using Taylor expansion. Smeers and De Wolf (2000) also uses an iterative solution process based on a modified simplex algorithm. An MIP approach for the stationary case is also applied in Möller (2004) and Martin et al. (2006). Here the nonlinearities are approximated

via SOS techniques. A global optimum is received in dependence on the approximation accuracy. For the more challenging transient case much less is known.

In Vostrý (1993) and Ehrhardt and Steinbach (2005) nonlinear optimization methods can be found whereas the first uses sub-gradient techniques and the latter SQP-methods. Note that these approaches do not include combinatorial conditions and just guarantee local optimality. In Sekirnjak (1998, 1999) the problem is formulated as a mixed integer program and is iteratively solved using sequential linear programming similar to Pratt and Wilson (1984). Westphalen (2004) concentrates on stochastical aspects of TTO and uses a coarse approximation of the nonlinearities. A linear model for transient gas flow is presented in Nowak and Westphalen (2003) where a simple line-pack model for pipes is developed and numerically tested.

In summary, for the steady state a lot of approaches are available in the literature. To the best of our knowledge, for the transient case, sequential solution methods are used that only guarantee local optimality, either combinatorial aspects are neglected or coarse approximations for the nonlinearities are applied. In the following we present a simulated annealing algorithm that includes time-dependent, combinatorial as well as nonlinear aspects and is applicable to general graph structures.

2 A mathematical model

We formulate the TTO problem as a mixed integer nonlinear problem (MINLP). The gas network is modeled via a directed graph $G = (V, E)$. The set of edges E can be divided into three kinds of components, also called segments, the set E_P of pipes, the set E_C of compressors, and the set E_V of valves. Further on we have special kinds of valves $E_R \subset E_V$, so called control valves, having the additional possibility to reduce gas pressure. Connections, a subset of the pipes $E_A \subset E_P$, can be viewed as short pipes where pressure loss can be neglected. The set of nodes V consists of the set of sources (gas delivering points) $S \subset V$, the set of sinks (consumers) $U \subset V$, and intersection points of segments. We assume the graph to be directed since we do not allow back-flow in pipes.

In the following we describe the variables and constraints of the model. In most cases we restrict to a verbal description since a detailed model would go beyond the scope of this paper. For an exact mathematical formulation we refer to Martin et al. (2006) and Möller (2004) in the stationary case and Mahlke (2005) and Moritz (2006) in the transient case.

Variables

Let $T \in \mathbb{N}$ denote the number of time periods (in our case hours), in which the planning horizon is divided. All variables receive a time index t for each $t \in \mathbb{T} := \{1, \ldots, T\}$. We introduce a variable $q_e^t \in \mathbb{R}_+$ describing the gas flow for each valve or compressor $e \in E_V \cup E_C$. For each pipe $e = vw \in E_P$ the two flow variables $q_{e,v}^t, q_{e,w}^t \in \mathbb{R}_+$ nominate the gas flow at the beginning and at the end of pipe e. For $v \in V$, $p_v^t \in \mathbb{R}_+$ describes the gas pressure. For each compressor $e \in E_C$, $f_e^t \in \mathbb{R}_+$ reflects the fuel gas consumption of the machine. We get constant upper and lower bounds for all these continuous variables. Finally, decision variables $s_e^t \in \{0, 1\}$, $e \in E_C \cup E_V$, indicate if a compressor or a valve is switched on or off.

Time-independent conditions
Now we describe conditions which must be considered for each time step $t \in \mathbb{T}$ of the planning horizon.

For *pipes*, there are no time-independent conditions as the most important pipe constraints arise because of the gas dynamics which have effect on gas flow and pressure. In case of a connection, a pipe without pressure loss, gas dynamics can be neglected and we get constant flow and pressure in it.

A *valve* is a controllable element which can be opened or closed. It is also found in connection with a compressor as a so called bypass valve. A bypass valve is open iff the compressor is not operating. The following constraints model the situation of a compressor $e = vw \in E_C$ and its bypass valve $b(e) \in E_V$

$$\begin{aligned} & s_e^t + s_{b(e)}^t = 1 \\ & dp_e^{\max} s_{b(e)}^t - p_v^t + p_w^t \leq dp_e^{\max} \\ & p_v^t - p_w^t \leq 0, \end{aligned}$$

where $dp_e^{\max}$ signifies the maximal pressure difference between beginning and end node. The pressure at the beginning and the end of an ordinary valve must be the same, whereas for control valves pressure can be reduced within some technical limits.

A *compressor* compensates for the pressure loss in pipes resulting in costs reflected by the fuel gas consumption. For a compressor $e = vw \in E_C$ the fuel gas f_e^t is given by the nonlinear function

$$f(p_v^t, p_w^t, q_e^t) = \gamma \left(\left(\frac{p_w^t}{p_v^t} \right)^{\frac{\kappa-1}{\kappa}} - 1 \right) q_e^t, \tag{1}$$

where γ, κ are constants. Note that (1) is neither convex nor concave.

For each *node* $v \in V \setminus \{S, U\}$ the first law of Kirchhoff must hold which ensures flow balance in a node

$$\begin{aligned} & \sum_{e \in \delta^-(v) \setminus E_P} q_e^t + \sum_{e \in \delta^-(v) \cap E_P} q_{e,v}^t - \sum_{e \in \delta^-(v) \cap E_C} f_e^t \\ & \quad = \sum_{e \in \delta^+(v) \setminus E_P} q_e^t + \sum_{e \in \delta^+(v) \cap E_P} q_{e,v}^t, \end{aligned} \tag{2}$$

where $\delta^+(v)$ and $\delta^-(v)$ denote the set of outgoing and ingoing segments, respectively. Note that the fuel gas consumption must be regarded in this equation if $e \in \delta^-(v)$ corresponds to a compressor. For sources and sinks we have additional flow bounds reflecting consumer behavior and supply contracts.

Gas dynamic in pipes

The gas transport in a pipe is described by a system of partial differential equations, i.e., the continuity equation, the momentum equation, and the energy equation as well as the thermodynamic state equation of gas. Since German pipes are at least 1 m beneath the ground with nearly constant temperature, general simulation models do not take the energy equation into account (see Sekirnjak 1999; Záworka). Therefore, we neglect it and assume the temperature T to be constant.

The continuity equation describes the influence of the alteration of the gas flow to the alteration of the gas density. The momentum equation specifies the sum of all forces on the gas molecules. The system of these equations is given by

$$A\,\frac{\partial \rho}{\partial t} + \rho_0\,\frac{\partial q}{\partial x} = 0, \tag{3}$$

$$\frac{\partial p}{\partial x} + g\rho\frac{\partial h}{\partial x} + \frac{\lambda |v| v}{2D}\rho + \frac{\rho_0}{A}\frac{\partial q}{\partial t} + \frac{\partial(\rho v^2)}{\partial x} = 0, \tag{4}$$

where A is the area of the cross-section of the pipe, ρ the gas density, ρ_0 the norm gas density, which is the density of the gas volume at norm pressure p_0 and norm temperature T_0 resulting from the state equation of gas, and q the gas flow. In the momentum equation g denotes the acceleration constant due to gravity, $\frac{\partial h}{\partial x}$ the slope of the pipe, λ is the pipe friction value, $v = \frac{\rho_0}{A}\frac{q}{\rho}$ the gas velocity, and D the diameter of the pipe. Since we assume the pipe to be horizontal we neglect the second term. The state equation of gas reads

$$\rho = \frac{\rho_0 z_0 T_0}{p_0}\,\frac{p}{z(p)T}, \tag{5}$$

where z is the compressibility factor characterizing the non-ideal behavior of gas (for ideal gas $z \equiv 1$ holds) and z_0 is the norm compressibility factor.

After insertion of Eq. (5) in the partial differential equations (3) and (4), we discretize them in space and time. For the space discretization, we use the grid consisting just of the beginning and the end of the pipe. For $e = vw \in E_P \setminus E_A$ this results in

$$\frac{q_w^t - q_v^t}{L} + A\frac{z_0 T_0}{p_0 T}\left(\frac{p_w^t}{z(p_w^t)} - \frac{p_w^{t-1}}{z(p_w^{t-1})}\right) = 0, \tag{6}$$

$$\frac{p_w^t - p_v^t}{L} + \frac{\lambda}{2D}\frac{\rho_0 p_0 T}{A^2 z_0 T_0}\frac{(q_w^t)^2 z(p_w^t)}{p_w^t} + \frac{\rho_0}{A}\left(q_w^t - q_w^{t-1}\right)$$

$$+\frac{\rho_0 p_0 T}{A^2 z_0 T_0}\frac{\frac{(q_w^t)^2 z(p_w^t)}{p_w^t} - \frac{(q_v^t)^2 z(p_v^t)}{p_v^t}}{L} = 0. \tag{7}$$

These two nonlinear equations are integrated in our model for each pipe $e \in E_P \setminus E_A$ and for each $t \in \mathbb{T} \setminus \{1\}$.

Boundary conditions

Moreover, we are given an initial state for the gas network, i.e., values of all flow and pressure variables at the beginning of the planning horizon. Based upon this initial state optimization for the succeeding time steps is made. Finally, a terminal condition is required which guarantees operational availability after the considered time horizon. We require that the total gas volume at the end of the considered time horizon is at least as large as that at the beginning. Otherwise the optimization process would result in very low pressure and flow values for the last time step, since it is cheaper to pump the gas network empty than to transport the gas flow from the sources.

Objective function
The objective function reads

$$\sum_{t\in\mathbb{T}} \sum_{e\in E_C} f_e^t, \tag{8}$$

which is to minimize the sum of the fuel gas consumption of the compressors for all time steps.

Altogether we get a complex model including nonlinearities and binary variables. Practical instances of this MINLP cannot be solved by current general purpose algorithms. It is even very difficult to determine a feasible solution for the TTO problem.

3 The simulated annealing algorithm

The simulated annealing (SA) algorithm is a meta-heuristic. It was originally developed for solving large combinatorial optimization problems and uses local search. It randomly accepts solutions with increasing objective function value. Thus, SA can overcome local minima and the dependence on the initial solution is marginal, but it stays flexible and robust (Aarts and Korst 1989).

The original idea of simulated annealing for solving large combinatorial optimization problems was independently introduced by Kirkpatrick et al. (1983) and Černý (1985). As implied by the name the simulated annealing algorithm is based on the analogy between the physical process of annealing liquids to the thermal equilibrium (solid phase) and the problem of finding the solution of combinatorial optimization problems.

The simulated annealing algorithm is widely discussed in the literature see for instance van Laarhoven and Aarts (1987), Aarts and Korst (1989), Reeves (1993), Osman and Kelly (1996), and Nemhauser and Wolsey (1988). It can be seen as a sequence of consecutively executed Metropolis algorithms (Metropolis et al. 1953) (see Algorithm 1, step 5 to 12). After each iteration of the Metropolis algorithm the control parameter $\mathcal{T}$ is reduced. Here, we give a description of the algorithm (Mahlke 2005).

Algorithm 1 Simulated Annealing

1: Initialize control parameter $\mathcal{T}$
2: Find initial solution S
3: Calculate $c(S)$
4: **repeat**
5: **repeat**
6: Randomly generate neighbor S' of S
7: Set $\Delta = c(S') - c(S)$
8: Choose $\theta \in (0, 1)$ randomly
9: **if** $\Delta < 0$ or $\theta < e^{-\Delta/\mathcal{T}}$ **then**
10: Set $S = S'$
11: **end if**
12: **until** Equilibrium criterion is fulfilled
13: Decrement control parameter $\mathcal{T}$
14: **until** Specified stop criterion is fulfilled
15: return the best solution found

The key point is that it also accepts worse solutions with a certain probability, which diminish with decreasing $\mathcal{T}$ (see Algorithm 1, step 8 and 9). This stochastic aspect should avoid termination in local minima. Therefore this algorithm is also called *stochastic hill-climber* (Michalewicz and Fogel 2000).

By means of the theory of Markov chains theoretical (asymptotic) convergence can be proven (van Laarhoven and Aarts 1987; Aarts and Korst 1989). Thus the algorithm can be viewed as a global optimization algorithm if an infinite number of transitions is allowed. Since this conclusion is not useful for practical applications, a finite time approximation must be developed (van Laarhoven and Aarts 1987; Aarts and Korst 1989). In the literature finite time bounds for SA can also be found (Nolte and Schrader 2000). The number of necessary steps, however, exceeds the cardinality of the solution space, hence it would be more reasonable to determine the optimal solution by complete enumeration.

Before implementing SA for the solution of a special problem, a lot of decisions must be made which can be divided in generic and problem specific ones (van Laarhoven and Aarts 1987; Reeves 1993). Generic decisions comprise parameters of the annealing algorithm itself. These are initialization and decrement of the control parameter $\mathcal{T}$, specification of the number of steps until the equilibrium condition is fulfilled and a stop criterion. Together these parameters build the so called *cooling schedule*. The second, problem specific class includes the characterization of feasible solutions, specification of the cost function, generation of an initial solution and the definition of the neighborhood structure.

As already mentioned, the presented SA algorithm is a method for solving combinatorial optimization problems. In that case it is often intuitive to specify a neighborhood structure. Because of its general and flexible form SA can easily be adapted to other optimization problems, since for example no special properties as differentiability or convexity are imposed on the objective function or the constraints.

Due to successful implementations of the SA algorithm in the field of combinatorial optimization, its practicability for problems with continuous variables was investigated. Approaches for global optimization in case of an n-dimensional function defined on a bounded subset can be found for instance in Bohachevsky et al. (1986), Dekkers and Aarts (1991). Dekkers and Aarts (1991) also proves convergence of the algorithm in analogy to the classical SA for combinatorial optimization problems.

Wah and Wang (1999) describe a modified SA algorithm to optimize functions with continuous variables subject to equality and inequality constraints. All constraints are relaxed by means of Lagrange multipliers and the resulting problem is solved using so called constrained simulated annealing. Note that if the optimization problem has numerous constraints one gets a lot of additional variables. In Corana et al. (1987) and Wah and Chen (2000) further modifications of SA can be found to tackle (constrained) global optimization of functions with continuous variables.

All these approaches provide a basis for solving the TTO problem. Note that our problem comprises numerous integer as well as continuous variables. Thereto we have to combine different ideas of the cited literature.

4 Simulated annealing for TTO

In order to adjust the simulated annealing algorithm to the TTO problem, we have to make a couple of problem specific and generic decisions. The latter concerns the design of an adequate cooling schedule. The decisions tailored to our problem comprise constraint-handling and solution characterization respectively, definition of a cost function, design of a neighborhood structure, step size selection and initial solution generation. The following discussion is gathered around these decisions.

Constraint-handling
Like in most real-world problems our problem contains a lot of constraints which restrict the solution space. The question is how to handle these constraints in order to receive feasible solutions. Basically there are two alternative approaches dealing with constraints, see Michalewicz and Fogel (2000). The first method only operates on feasible solutions and all infeasible solutions are rejected. Thus, the search space is not extended supplementary. A disadvantage is the difficulty of finding feasible neighbors in each iteration step. The second approach deals with both feasible and infeasible solutions and therefore simplifies the latter problem. The new arising task of evaluating infeasible solutions can be handled by adding a penalty term $Q(x)$ to the objective function in order to reduce the quality of an infeasible solution x.

Since it is very complicated to find a feasible solution for the TTO problem we apply the latter approach. In Michalewicz and Fogel (2000) two commonly known penalty methods are provided, a static and a dynamic one. For a better understanding in the following we abbreviate a solution (p, q, f, s) consisting of pressure, flow, fuel gas consumption, and decision variables by x. Further on we denote the equations and inequalities presented in Sect. 2 by $h_j(x)$ and $g_j(x)$, respectively. Both penalty methods are based on the following definition of $v_j(x)$ for a solution x which is a measure for the violation of the j-th constraint:

$$v_j(x) = \begin{cases} \left|h_j(x)\right| & \text{if } 1 \le j \le q \\ \max\left\{0, g_j(x)\right\} & \text{if } q+1 \le j \le m, \end{cases}$$

where $q \in \{1, \ldots, m\}$ is the number of equations and $j \in \{1, \ldots, m\}$. The penalty function for the static method is

$$Q(x) = \sum_{j=1}^{m} R_j v_j^2(x),$$

where R_j is a constant imposed for the violation of constraint j. To overcome the problem of determining good parameters R_j, which are necessary to find feasible solutions, penalty functions with dynamic aspects are applied. As a result of including the iteration step n to the function the penalty term is dynamically increased during execution. For the n-th iteration the penalty function is calculated by

$$Q(x) = \sum_{j=1}^{m} (C_j n)^{\alpha} v_j^2(x),$$

where C_j and α are positive constants. While testing the SA algorithm applying these two methods to our problem there arise difficulties in finding a feasible

solution, see Mahlke (2005). Thus, we developed a combination of both methods especially adapted to our problem. Besides the minimization of the fuel gas consumption (8) the feasibility of the solution is the primary objective of the algorithm. In this context we consider a solution to be feasible, if the maximal absolute violation of the relaxed constraints is lower than a given accuracy ε. The underlying idea of this combined method is the additional dynamic penalty of violations greater than ε. For TTO the penalty function is defined as

$$Q(x) = \sum_{j=1}^{m} P_j v_j^2(x), \tag{9}$$

where

$$P_j = \begin{cases} R_j & \text{if } v_j(x) \leq \varepsilon \\ R_j + (C_j n)^\alpha & \text{if } v_j(x) > \varepsilon. \end{cases}$$

Having presented the different penalty functions, we apply this approach to our problem. As good parameter settings for the penalty functions might be difficult to find we neglect only some constraints which pose most of the problems to us. Therefore, we relax the discretized continuity (6) and momentum (7) equations for each pipe of the gas network, as basically they connect single time steps and flow and pressure variables. The remaining constraints have to be fulfilled in each iteration. Once the penalty functions $Q_C(x)$ and $Q_M(x)$ of the form (9) for the continuity and momentum equations are introduced, the problem reads

$$\begin{aligned} &\min && f(x) + Q_C(x) + Q_M(x) \\ &\text{s.t. } x \text{ satisfies} && \bullet \text{ lower and upper bounds for pressure in nodes} \\ &&& \bullet \text{ lower and upper bounds for flow in segments} \\ &&& \bullet \text{ flow balance in nodes (Kirchhoff's first law), see (2)} \\ &&& \bullet \text{ supplier and consumer behavior for sources and sinks} \\ &&& \bullet \text{ compressor constraints, see (1)} \\ &&& \bullet \text{ switching constraints for valves and compressors} \\ &&& \bullet \text{ terminal condition for the last time step,} \end{aligned} \tag{10}$$

where $f(x)$ denotes the original objective function, i.e., the fuel gas consumption (8).

Neighborhood structure

The choice of a suitable neighborhood structure is, next to the cooling schedule, one of the most important parts of the algorithm. Only an efficient neighborhood structure leads to a powerful search. Since the problem contains continuous and integer variables, the definition of a neighborhood structure is not as intuitive as for combinatorial problems.

Our key idea of the neighborhood generation is a small perturbation of flow and pressure variables in the segments and nodes. At each iteration of the algorithm either a flow variable q_e^t, $q_{e,v}^t$ or $q_{e,w}^t$ of a segment $e = vw \in E$ at $t \in \mathbb{T} \setminus \{1\}$ is variated or a pressure variable p_v^t of a node $v \in V$ at time step $t \in \mathbb{T} \setminus \{1\}$ is altered. Due to the relaxation of the continuity and momentum equations each time

step can be treated separately and flow and pressure variables can be determined independently for each time step with the exception of the terminal condition, which couples the first and last time step.

We define two sets $N_{\text{pressure}}(x)$ and $N_{\text{flow}}(x)$ for a given solution x and denote x' to be a neighbor of x if $x' \in N(x) := N_{\text{pressure}}(x) \cup N_{\text{flow}}(x)$.

The alteration of a flow variable at $t \in \mathbb{T} \setminus \{1\}$ often results in a violation of constraints which concern other flow variables in the same time step. This holds for pressure variables analogously. In order to generate a feasible neighbor for the relaxed problem (10) we use so called *repair procedures* to adapt the neighbor to the violated constraints or to return that the generated neighbor is infeasible.

In detail, an element of $N_{\text{flow}}(x)$ is generated in the following manner. The generation mechanism selects randomly a segment $e \in E$ and a time step $t \in \mathbb{T} \setminus \{1\}$. For $e \in E_C \cup E_V \cup E_A$ we modify the corresponding flow variable q_e^t by increasing or decreasing its value by Δq_e^t, where Δq_e^t denotes the step size. In case of choosing a pipe $e = vw \in E_P \setminus E_A$ to which a flow variable at the beginning and a flow variable at the end are assigned we decide randomly between $q_{e,v}^t$ and $q_{e,w}^t$ and alter this flow variable analogously. In case of a capacity violation the generation terminates.

Once the flow variable is set there are basically two constraints that can be violated, the flow balance equation (2) in the initial node v and the flow balance equation in the end node w of edge e at time t. Here the approach to handle these violations is to adjust the flow variables concerning node w in flow direction by the recursive procedure *adjustFlowForward*() and the flow variables corresponding to node v in reverse flow direction by the recursive procedure *adjustFlowBackward*(). Both procedures are based on the specific properties of each segment type. Basically the segments can be subdivided into two groups: *free* segments and *fixed* segments. A segment belongs to the first group, if a modification of the flow variable at the beginning of the segment does not effect the flow variable at the end and reversely. All other segments are called fixed. The assignment of the segment depends on the segment specific constraints. In our model compressors, valves, control valves and connections are fixed, whereas pipes are free segments due to the relaxed momentum and continuity equations. Briefly the *adjustFlowBackward*() procedure works in the following way. At the beginning it is checked whether v is a source. If so, the procedure terminates. Otherwise it selects randomly one of the ingoing edges of v, say $e' = uv$. In case of a fixed segment the corresponding flow variable is modified according to the balance equation (2) in node v and the procedure is recursively applied to node u. Otherwise, in case of a free segment, that is a pipe. The corresponding flow variable $q_{e',v}^t$ is adjusted and the procedure terminates. Unless no capacity constraint is violated the procedure returns "feasible". The *adjustFlowForward*() procedure works analogously. To illustrate the flow neighbor generation we give the following example (see Fig. 1).

Let $e_3 \in E$ be the randomly chosen segment which is a connection. Since we consider a fixed, randomly selected time step t, in the following we omit this index. At the beginning the flow variable of the connection is modified by the generated step length Δq_{e_3}. Assume that no capacity constraint is violated. In order to adjust the resulting violation of the Kirchhoff law in the initial node v the *adjustFlowBackward*() procedure is called. Selecting randomly the open valve e_2 out of the ingoing edges of node v, the associated variable q_{e_2} is adjusted. Note that

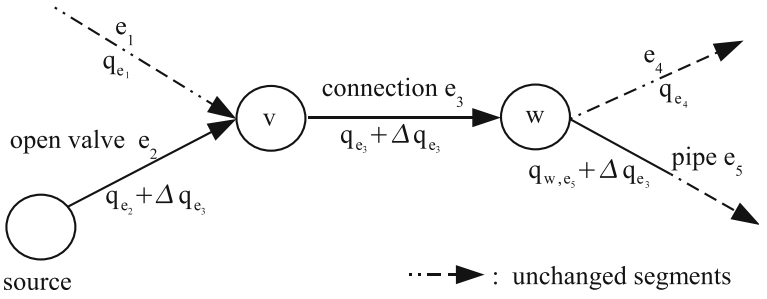

Fig. 1 Flow neighbor

a valve is a fixed segment, hence recursion begins. Since the initial node of edge e_2 is a source, the procedure terminates successfully, granted that no capacity constraint is violated. We proceed with the *adjustFlowForward*() procedure in node w. Suppose that the pipe e_5 is selected randomly. Because of e_5 being a free segment, the corresponding variable $q_{e_5,w}$ is adapted and the procedure terminates. Verifying the capacity constraints, a feasible flow neighbor is generated.

To generate an element of $N_{\text{pressure}}(x)$ we proceed similarly to the flow neighbor generation. The main difference lies in the fact that pressure variables are nodal. Thus, the generation mechanism selects randomly a node $v \in V$ and a time step $t \in \mathbb{T} \setminus \{1\}$. Subsequently the corresponding pressure variable p_v^t is altered by decreasing or increasing its value by Δp_v. The resulting constraint violations are repaired by the *adjustPressure*() procedure. Briefly, this procedure is based on the same idea as the *adjustFlowBackward*() procedure, but works independently from the flow direction. Again, we differentiate between fixed and free segments, this time with respect to pressure. Here open valves and connections are fixed, since pressure equality is required. Whereas pipes, compressors, control valves and closed valves are free segments. Note that these segments are able to deal with different values of the variables in the initial and end node. In order to adjust the neighbor to the violated constraints the recursive *adjustPressure*() procedure is called for every fixed segment incident with node v.

Additionally the algorithm provides the possibility of closing or opening the operable segments, i.e., compressors and valves. If in the scope of neighborhood generation the fuel gas consumption of an open compressor falls below a specialized lower bound the compressor can be closed with a certain probability and with respect to switching constraints. Accordingly the same holds for an open valve, if the corresponding flow variable reaches its lower bound. Further on a compressor or a valve can be opened if its associated flow variable is increased during neighborhood generation.

Step size selection

A special task arising for optimization problems involving continuous variables is the determination of an appropriate step size. This choice has a major effect on the accuracy of the solution. Generally there is a tradeoff between accuracy and speed. To alleviate this problem we use the approach presented by Miki et al. (2003). We determine Δx_i by applying an uniform distribution and a fixed neighborhood range

R_i for each variable x_i. Thus, the step size is generated by the following equation

$$\Delta x_i = r \cdot R_i, \quad (-1 \leq r \leq 1),$$

where r denotes a random variable. In general the neighborhood ranges R_i are application dependent. In this context we need neighborhood ranges R_p and R_q for pressure and flow variables which are determined according to several test runs, see Sect. 5.

Initial solution generation
To generate an initial feasible solution for the relaxed problem (10), we use the initial state of the gas network. The variable values of this state are assigned to the corresponding variables at the time steps $t \in \mathbb{T} \setminus \{1\}$. Thus, only transient constraints are violated. The latter can be repaired using the procedures developed for the neighborhood generation.

Cooling schedule
As already mentioned, a cooling schedule is a set of parameters, specified by an initial value and a decrement function of the control parameter $\mathcal{T}$, a finite number of iterations for each value of the control parameter and a stop criterion. According to these parameters the probability of accepting worsening moves is controlled. The problem of determining a good cooling schedule is addressed in several papers, e.g. see van Laarhoven and Aarts (1987) and Dekkers and Aarts (1991). In this section we present the schedules used for TTO.

- *Initial value of the control parameter $\mathcal{T}$:* An initial value is chosen so that at the beginning approximately all transitions are accepted, i.e., the ratio between the number of accepted transitions and the number of generated ones is close to 1. We follow Dekkers and Aarts (1991) who proposes the following empirical rule: Suppose a given number of trials m_0 is generated under the assumption that all transitions are accepted. Let m_1 and m_2 denote the number of improving and worsening trials respectively ($m_1 + m_2 = m_0$). The average increase in cost is described by $\bar{\Delta} f^+$. For a given acceptance ratio $\chi_0 \in (0, 1)$ the initial value $\mathcal{T}_0$ is calculated from the following expression

$$\mathcal{T}_0 = \bar{\Delta} f^+ \left(\ln \frac{m_2}{m_2 \chi_0 + (1 - \chi_0) m_1} \right)^{-1}. \tag{11}$$

- *Finite number L of iterations for each value of the control parameter:* This number L regulates how thoroughly the search space is explored. It should be large enough to allow the algorithm to investigate the neighborhood of a given solution in all directions. Dekkers and Aarts (1991) determines L depending on the problem dimension n and a constant L_0 by the following expression

$$L = L_0 \cdot n.$$

  For our problem we determine L to be proportional to the number of time steps T with $L_0 = 500$.
- *Decrement function of the control parameter:* We present two alternative functions: geometric and adaptive. The geometric function is a commonly used decrement rule as it simply requires a constant $\alpha \in (0, 1)$, see van Laarhoven

Table 1 Three test instances

Test instance	Pipes	Number of compr.	Valves	Length of all pipes
Network 1	11	3	5	920
Network 2	20	3	4	1,200
Network 3	31	15	29	2,200

and Aarts (1987). The control parameter is decreased according to the following rule

$$\mathcal{T}_k = \alpha \mathcal{T}_{k-1},$$

where $\mathcal{T}_k$ denotes the value of the control parameter in the k-th inner loop. An alternative adaptive approach is given in Aarts et al. (1986). In contrast to the former approach adaptive decrement functions use information gathered during execution. Here, the reduction multiplicator is recalculated in each iteration step using the standard deviation $\sigma(\mathcal{T}_{k-1})$ of the objective function in the $(k-1)$-th inner loop and some parameter δ:

$$\mathcal{T}_k = \mathcal{T}_{k-1} \left(1 + \frac{\ln(1+\delta)\mathcal{T}_{k-1}}{3\sigma(\mathcal{T}_{k-1})}\right)^{-1}. \tag{12}$$

The parameter δ influences the velocity of decreasing the control parameter.

- *Stop criterion:* The algorithm terminates if either the control parameter reaches a predefined final value $\overline{\mathcal{T}}$ or if there is no significant improvement for a fixed number n of iterations, see van Laarhoven and Aarts (1987). Here, we used $\overline{\mathcal{T}} = 0.001$ and $n = 100{,}000$. Additionally, the algorithm stops if the maximum absolute violation of the relaxed constraints falls below a given tolerance value ε, since our primary aim is to find a feasible solution. Note that in the latter case the algorithm finds a feasible solution for the original problem described in Sect. 2 since we allow minor violations of the discretized continuity and momentum equations.

5 Computational results

The simulated annealing algorithm was implemented in C and the computations were done on a 1 GHz Pentium III processor with 1 GB main memory.

For the computational tests we consider three test instances provided by the German gas company E.ON Ruhrgas AG, see Table 1.

Figure 2 shows the smallest test instance, which consists of three compressors (named VdA, VdB and VdC), 11 pipes with a total length of 920 km, five valves, two sources (Qu1, Qu2) and three sinks (Ab1, Ab2, Ab3).

Since SA has a stochastic component it is reasonable to run the algorithm several times applying different random numbers. In order to receive meaningful results, here the outcomes are average values of ten independent runs.

Based on various test runs we integrated the following methods in the SA algorithm. We applied the adaptive decrement function (12) using the parameter

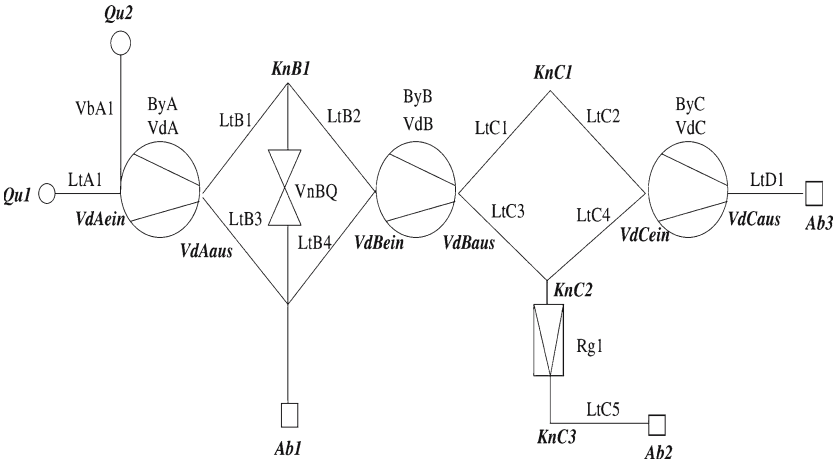

Fig. 2 A test instance

Table 2 Computational results depending on accuracy parameter and step size range

ε	R_p	R_q	Fuel	Iter.	Sec.	Succ. rate
0.1	0.01	1	36.98	193,833	26.73	9/10
	0.05	5	36.64	68,700	11.52	10/10
	0.25	25	37.40	91,500	14.00	8/10
0.05	0.01	1	36.21	310,875	43.08	8/10
	0.05	5	36.65	81,300	12.80	10/10
	0.25	25	37.58	184,071	24.97	7/10
0.01	0.01	1	36.47	306,562	41.86	8/10
	0.05	5	36.56	236,333	31.91	9/10
	0.25	25	37.11	1,044,000	130.66	6/10

$\delta = 20$. For the initial value of the control parameter we used Dekkert's and Aart's approach, see Eq. (11). The calculation of the penalty function was done according to the combined penalty function (9), where $R_j = 10$ and $C_j = 0.0005$ for all j and $\alpha = 2$. The number of iterations L for each value of the control parameter $\mathcal{T}$ was computed by the expression $L = 500(T - 1)$, where T denotes the planning horizon.

Table 2 gives an impression of the behavior of the algorithm for different accuracy parameters ε, specified in the first column, while varying the step size ranges R_p and R_q, shown in column two and three. We tested the algorithm for network 1 with planning horizon $T = 4$. In the columns Fuel, Iter., and Sec. the fuel gas consumption, the total number of iterations and the running time can be found. As already mentioned these three values are average values of ten independent runs. The last column shows the success rate m/n of the algorithm, which means that in m out of n runs the algorithm finds a feasible solution to the original problem.

Generally, we notice only minor differences of fuel gas consumption between the several test runs. In contrast the solution time varies considerably. The table illustrates the flexibility of the variable step size selection with respect to different accuracies. For every considered ε the best results are obtained using $R_p = 0.05$

Table 3 Computational results on various time periods for network 1

T	Fuel	Iter.	Sec.	Succ. rate
3	27.13	41,900	6.02	10/10
6	56.05	118,888	24.99	9/10
12	115.57	314,285	95.11	7/10
24	235.73	660,611	299.19	9/10

Table 4 Computational results on various time periods for network 2

T	Fuel	Iter.	Sec.	Succ. rate
3	14.98	109,250	14.62	8/10
6	31.07	381,944	71.29	9/10
12	66.11	1,281,500	304.17	8/10
24	139.09	2,870,400	944.13	10/10

Table 5 Computational results on various time periods for network 3

T	Fuel	Iter.	Sec.	Succ. rate
3	5.11	37,222	4.15	9/10
6	10.21	67,857	9.51	7/10
12	20.46	132,916	24.11	10/10
24	40.94	287,500	76.48	10/10

and $R_q = 5$. Furthermore, there is a clear dependency between running time and selected accuracy. With decreasing ε the running time increases moderately.

We are interested in the flexibility of the algorithm regarding the complexity of the tested network. Table 3 shows the results of our testing calculations using several numbers of time steps T, given in the first column. We applied the same step size selection, penalty function, decrement function, and parameter setting as in the previous test runs. Further on we used an accuracy of $\varepsilon = 0.1$ and step size ranges $R_q = 5$ and $R_p = 0.05$. The running time of the algorithm seems acceptable for all number of time steps we have been investigated. Nevertheless we observe a notable raise of running time with increasing T, due to the growing dimension of the problem.

Tables 4 and 5 show the computational results on the second and third network. Network 2 consists of three compressors and 20 pipes with a total length of 1,200 km and network 3 contains 15 compressors and 31 pipes with a total length of 2,200 km, see Table 1. All settings used in the algorithm are taken over, except for the control parameter δ of the decrement function. In order to obtain feasible solutions, we decreased its value from 20 to 5 for the third test instance. We see that the running times shown in Table 4 are considerably longer than those shown in Table 3 even though the complexity of the test instances only increases slightly. Among others this is due to good initial solutions generated for the computations of the first network.

Concerning the results given in Table 5 the running times are surprisingly small even though network 3 is much bigger than network 1 and 2. Again this observation can be attributed to good initial solutions generated for the third network.

Recapitulating these results, the algorithm is able to find feasible solutions for all test instances considering up to 24 times steps in very fast running time.

6 Conclusions

In this paper, we adapted the idea of simulated annealing to the problem of transient technical optimization (TTO). Since the gas network optimization problem contains numerous integer and continuous variables, we had to combine different approaches from literature resulting in a new algorithm. We relaxed the discretized continuity and momentum equations which mainly connect the single time steps as well as pressure and flow variables. Then, we developed a cost function combining static and dynamic penalty functions. Furthermore, we propose a flow and pressure neighborhood structure for the relaxed problem. An adequate cooling schedule was defined following proposals of the literature.

Altogether we get a flexible algorithm. Moreover, just minor modifications of parameters are necessary to tackle gas networks of different dimension. The proposed SA algorithm yields feasible solutions for TTO in very fast running times.

A first step toward transient optimization of gas networks is done. The great challenge determining a provably good or optimal solution for time-dependent gas networks stays open. The next step would be to integrate this heuristic solution as upper bound in a branch-and-cut algorithm.

Acknowledgements We are thankful to our industry partner E.ON Ruhrgas AG for the provided test instances.

References

Aarts E, Korst J (1989) Simulated annealing and Boltzmann machines. A stochastic approach to combinatorial optimization and neural computing. Wiley, New York

Aarts EHL, de Bont FMJ, Habers EHA, van Laarhoven PJ (1986) Parallel implementations of the statistical cooling algorithm. Integr VLSI J 4:209–238

Bohachevsky IO, Johnson ME, Stein ML (1986) Generalized simulated annealing for function optimization. Technometrics 28(3):209–217

Borraz-Sánchez C, Ríos-Mercado RZ (2005) A hybrid meta-heuristic approach for natural gas pipeline network optimization. In: Blesa MJ, Blum C, Roli A, Sampels M (eds) Hybrid Metaheuristics: Second International Workshop, HM 2005, Barcelona, Spain, August 29–30. Springer, Heidelberg, pp 54–65

Carter R (1998) Pipeline optimization: dynamic programming after 30 years. In: Pipeline Simultaion Interest Group, URL: http://www.psig.org

Černý V (1985) Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm. J Optim Theory Appl 45:41–51

Corana A, Marchesi M, Martini C, Ridella S (1987) Minimizing multimodal functions of continuous variables with the "simulated annealing" algorithm. ACM Trans Math Softw 13:262–280

Dekkers A, Aarts E (1991) Global optimization and simulated annealing. Math Program 50: 367–393

Ehrhardt K, Steinbach M (2005) Nonlinear optimization in gas networks. In: Bock HG, Kostina E, Phu HX, Ranacher R (eds) Modeling, simulation and optimization of complex processes. Springer, Berlin, pp 139–148

Gopal VN (1979) Techniques to optimize fuel and compressor combination selection. In: American Gas Association Transmission Conference

Jenicek T (1993) Steady-state optimization of gas transport. In: Proceedings of the 2nd International Workshop SIMONE on Innovative Approaches to Modelling and Optimal Control of Large Scale Pipeline Networks, September 1993
Kirkpatrick S, Gelatt CD Jr, Vecchi MP (1983) Optimization by simulated annealing. Science 220:671–680
Králik J (1993) Compressor stations in SIMONE. In: Proceedings of the 2nd International Workshop SIMONE on Innovative Approaches to Modelling and Optimal Control of Large Scale Pipeline Networks, September 1993
van Laarhoven PJM, Aarts EHL (1987) Simulated annealing: theory and applications. D. Reidel, Dordrecht
Mahlke D (2005) Der Simulated Annealing Algorithmus zur transienten Optimierung von Gasnetzen. Master's thesis, Technische Universität Darmstadt
Martin A, Möller M, Moritz S (2006) Mixed integer models for the stationary case of gas network optimization. Math Program 105:563–582
Metropolis N, Rosenbluth AW, Rosenbluth MN, Teller AH, Teller E (1953) Equation of state calculations by fast computing machines. J Chem Phys 21(6):1087–1092
Michalewicz Z, Fogel DB (2000) How to solve it: modern heuristics. Springer, Heidelberg
Miki M, Hiroyasu T, Fushimi T (2003) Parallel simulated annealing with adaptive neighborhood determined by GA. IEEE Int Conf Syst Man Cybern 1:26–31
Möller M (2004) Mixed integer models for the optimisation of gas networks in the stationary case. PhD thesis, Darmstadt University of Technology
Moritz S (2007) A mixed integer approach for the transient case of gas network optimization PhD thesis, Darmstadt University of Technology (to appear)
Nemhauser GL, Wolsey LA (1988) Integer and combinatorial optimization. Wiley, New York
Nolte A, Schrader R (2000) A note on the finite time behavior of simulated annealing. Math Oper Res 25(3):476–484
Nowak MP, Westphalen M (2003) A linear model for transient gas flow. Technical Report STF38 S03601, SINTEF Industrial Management, Norway
Osman IH, Kelly JP (eds) (1996) Meta-heuristics: theory and applications. Kluwer, Dordrecht
Pratt KF, Wilson JG (1984) Optimization of operation of gas transmission systems. Trans Inst Meas Control 6(4):261–269
Reeves CR (ed) (1993) Modern heuristic techniques for combinatorial problems. Halstead Press (Wiley), New York
Sekirnjak E (1998) Mixed integer optimization for gas transmission and distribution systems. INFORMS Meeting, Seattle, October 1998. Lecture notes
Sekirnjak E (1999) Transiente Technische Optimierung (TTO-Prototyp). PSI Berlin, 1999. Technical Report
Smeers Y, De Wolf D (2000) The gas transmission problem solved by an extension of the simplex algorithm. Manage Sci 46:1454–1465
Vostrý Z (1993) Transient optimization of gas transport and distribution. In: Proceedings of the 2nd International Workshop SIMONE on Innovative Approaches to Modelling and Optimal Control of Large Scale Pipeline Networks, September 1993
Wah BW, Chen YX (2000) Optimal anytime constrained simulated annealing for constrained global optimization. In: Proceedings Sixth International Conference on Principles and Practice of Constraint Programming, pp 425–440
Wah BW, Wang T (1999) Constrained simulated annealing with applications in nonlinear continuous constrained global optimization. In: Proceedings of the 11th IEEE International Conference on Tools with Artificial Intelligence, pp 381–388
Westphalen M (2004) Anwendungen der stochastischen Optimierung im Stromhandel und Gastransport. PhD thesis, Universität Duisburg-Essen
Wright S, Somani M, Ditzel C (1998) Compressor station optimization. In: Pipeline Simulation Interest Group, Denver, Colorado, October 1998
Záworka J (1993) Project SIMONE—Achievements and Running Development. Institute of Information Theory and Automation, Czech Republik
Zimmer HI (1975) Calculating optimum pipeline operations. In: American Gas Association Transmission Conference