A Tree Decomposition Algorithm for Minimizing Fuel Cost in Gas Transmission Networks

Conrado Borraz-Sánchez[1], **Dag Haugland**[1]

[1]Department of Informatics, University of Bergen, Norway
{Conrado.Borraz-Sanchez, Dag.Haugland}@ii.uib.no

ABSTRACT

In this paper, we address the problem of computing optimal transportation plans of natural gas by means of compressor stations in pipeline networks. This non-linear (non-convex) problem takes into account two types of continuous decision variables: mass flow rate through each arc, and gas pressure level at each node. Compressors consume fuel at rates depending on flow and pressure, and the problem is to assign values to these variables such that the total fuel cost is minimized.
We propose a dynamic programming algorithm based on tree decomposition, which applies to a broader class of instances than currently available techniques can solve. Through computational experiments, we demonstrate that our algorithm is capable to solve several instances where previously suggested methods and commercially avialable solvers for non-linear optimization fail.

Keywords: Gas Transmission Network, Fuel Cost, Dynamic Programming, Tree Decomposition

1. INTRODUCTION

Natural gas has become one of the most important energy resources worldwide. Consequently, the volumes of gas flowing from the fields through transmission networks to the market have been increasing steeply during the past decades, and in parallel, a growing interest in reducing costs associated with pipeline gas transportation has been observed.

In this paper, the fuel cost minimization problem (FCMP) to transport natural gas in a general class of transmission networks is addressed. The FCMP involves two types of continuous decision variables: mass flow rate through each arc, and gas pressure level at each node. The problem can be described as follows: We need to move natural gas over large distances from several possible sources to different distribution centers through various devices including pipes and compressor stations. During the transmission, energy and pressure are lost, and the compressor stations installed in the pipeline system are crucial for keeping the gas moving. Consequently, fuel consumption associated costs are incurred at these stations. The problem is to determine a transportation plan on an existing network minimizing the total fuel cost, while meeting specified demand at the distribution centers.

1.1. Related work

An extensive literature on the FCMP has been published over the past 30 years. This includes applications of numerical simulations (see [10]), Dynamic Programming (DP) (see [6], [8] and [17]), gradient techniques (see [4]). and others. Most of these contributions are practically limited to pipelines networks with non-cyclic structures or to sparse cyclic networks, and have obtained a considerable success on such instances.

Several works based on successive reductions of the network (see [1], [2] and [3]), and graph theory and functional analysis (see [12]) have been developed with the promise to handle cyclic topologies. However, since these optimization approaches require a certain sparse network structure, their application is still in a development phase. The purpose of the current work is to present a solution approach that admits a more general network structure, and hence overcome the limitations of network reduction techniques.

The remainder of this paper is organized as follows. In Section 2, we define the problem in mathematical terms. In Section 3, we present a contemporary solution method, and point out a serious point of weakness. In Section 4, the tree decomposition based algorithm to solve the FCMP via DP is described. Our numerical results based on different computational experiments are shown in Section 5, where we compare our results to those obtained by alternative methods when applied to several network configurations. Finally, concluding remarks are given in Section 6.

2. PROBLEM DEFINITION

Let $G = (V, A)$ be a directed graph representing a gas transmission network, where V and A are the node and arc sets, respectively. Let V_v^+ and V_v^- denote the sets of out- and in-neighbors, respectively, of node $v \in V$, let $V_s \subseteq V$ be the set of supply nodes, $V_d \subseteq V$ the set of demand nodes, and let $A = A_c \cup A_p$ be partitioned into a set of compressor arcs A_c and a set of pipeline arcs A_p. That is, if $(u, v) \in A_c$ then $u, v \in V$ are the network node representing the input and the output units, respectively, of some compressor (u, v). An analogous interpretation is made for pipeline arcs $(u, v) \in A_p$.

Two types of decision variables are defined: Let x_{uv} denote the mass flow rate at arc $(u, v) \in A$, and let p_v denote the gas pressure at node $v \in V$. For each $v \in V$, we define the parameters net mass flow rate B_v and (lower and upper, respectively) pressure bounds P_v^L and P_v^U. By convention, $B_v > 0$ if $v \in V_s$, $B_v < 0$ if $v \in V_d$, and $B_v = 0$ otherwise. By the assumption that flow is conserved at the

978-1-4244-4136-5/09/$25.00 ©2009 IEEE

nodes, the decision variables are subject to the constraints $\sum_{u\in V_v^+} x_{vu} - \sum_{u\in V_v^-} x_{uv} = B_v$ for all $v \in V$. Constraints linking the pressure and flow variables are given for the arc sets A_c and A_p, and these are discussed next.

2.1. Compressor arc constraints

The variables that are manipulated in a compressor $(u, v) \in A_c$ in order to have the desired values of x_{uv}, p_u, and p_v are according to Wu et al. [18] compressor speed S_{uv}, volumetric inlet flow rate Q_{uv}, adiabatic head H_{uv} and adiabatic efficiency η_{uv}. As explained more detailed in e.g. [18], these relate to (x_{uv}, p_u, p_v) according to

$$H_{uv} = \alpha \left[\left(\frac{p_v}{p_u} \right)^m - 1 \right] \quad \forall (u,v) \in A_c \qquad (1)$$

$$Q_{uv} = \alpha m \frac{x_{uv}}{p_u} \quad \forall (u,v) \in A_c \qquad (2)$$

$$\frac{H_{uv}}{S_{uv}^2} = \phi^1 \left(\frac{Q_{uv}}{S_{uv}} \right) \quad \forall (u,v) \in A_c \qquad (3)$$

$$\eta_{uv} = \phi^2 \left(\frac{Q_{uv}}{S_{uv}} \right) \quad \forall (u,v) \in A_c \qquad (4)$$

where $m \in (0, 1)$ and $\alpha > 0$ are gas specific constants, and ϕ^1 and ϕ^2 are polynomial functions (typically of degree 3). The coefficients of ϕ^1 and ϕ^2 are assessed by applying least squares analysis to a set of selected data points. For each $(u, v) \in A_c$, Q_{uv} is subject to lower and upper bounds Q_{uv}^L and Q_{uv}^U, and we adopt a similar notation for bounds on the variables S_{uv}, H_{uv} and η_{uv}.

The fuel consumption cost is given by (see [18])

$$g_{uv}(x,p) = \frac{c x_{uv} \left[\left(\frac{p_v}{p_u} \right)^m - 1 \right]}{\eta_{uv}} \quad \forall (u,v) \in A_c,$$

where $c > 0$ is a monetary constant.

The *operating domain* of compressor $(u, v) \in A_c$ is the set $D_{uv} \subset \Re^3$ of value assignments to (x_{uv}, p_u, p_v) for which there exist values of $(Q_{uv}, S_{uv}, H_{uv}, \eta_{uv})$ satisfying (1)-(4) and the bounds $Q_{uv}^L \le Q_{uv} \le Q_{uv}^U$, $S_{uv}^L \le S_{uv} \le S_{uv}^U$, $H_{uv}^L \le H_{uv} \le H_{uv}^U$, and $\eta_{uv}^L \le \eta_{uv} \le \eta_{uv}^U$.

We assume that for all $(x_{uv}, p_u, p_v) \in D_{uv}$, there is a *unique* feasible $(Q_{uv}, S_{uv}, H_{uv}, \eta_{uv})$. This correspondence defines the desired transformation from feasible flow and pressure variable values (x_{uv}, p_u, p_v) to an estimate $g_{uv}(x, p)$ of the fuel cost.

2.2. Pipeline arc constraints

Following [18], the relation between pipeline flow and (sufficiently high) pressure in steady state networks can be written as $x_{uv}^2 = W_{uv}\left(p_u^2 - p_v^2\right)$, where $W_{uv} > 0$ is some constant depending on characteristics of the gas and the pipeline $(u, v) \in A_p$.

2.3. Mathematical model

For each node $v \in V$, we impose lower and upper pressure bounds P_v^L, and P_v^U, respectively. We confine our study to irreversible flow, and impose $x_{uv} \ge 0$ for all $(u, v) \in A$.

Summarizing the two last sections, the FCMP can then be formulated as follows:

$$\min \quad \sum_{(u,v)\in A_c} g_{uv}(x,p) \qquad (5)$$

s.t.:

$$\sum_{u\in V_v^+} x_{vu} - \sum_{u\in V_v^-} x_{uv} = B_v \quad \forall v \in V \qquad (6)$$

$$(x_{uv}, p_u, p_v) \in D_{uv} \quad \forall (u,v) \in A_c \qquad (7)$$

$$x_{uv}^2 = W_{uv}\left(p_u^2 - p_v^2\right) \quad \forall (u,v) \in A_p \qquad (8)$$

$$P_v^L \le p_v \le P_v^U \quad \forall v \in V \qquad (9)$$

$$x_{uv} \ge 0 \quad \forall (u,v) \in A \qquad (10)$$

3. SOLUTION METHODS

Several solution methods have been suggested for FCMP, including those by Ríos-Mercado et al. [11] and Borraz-Sánchez and Ríos-Mercado [1], which all follow the idea of Algorithm 1.

Algorithm 1 SolveFCMP

Step 1: Choose initial (feasible) flow
repeat
  Step 2: Optimize pressure while keeping the flow fixed
  Step 3: Optimize flow while keeping the pressure fixed
until flow does not change

With the risk of missing the global optimum, flow and pressure are determined separately in Steps 2 and 3, respectively. As we show next, this can be accomplished by focusing on only a subset of the variables.

3.1. Compressor network

In [12], it was shown that if $A_c = \emptyset$ then for any $B \in \Re^V$ there exists a unique solution to the set of equations defined by (6) and (8). That is, the flow assignment to A_p is unique (and infeasible if it violates (10)).

Let $V' \subseteq V$ consist of exactly one node from each of the connected components in the directed graph (V, A_p), and let $G^v = (V^v, A^v)$ denote the component (subgraph) to which $v \in V'$ belongs. By applying the result in [12] to G^v for any $v \in V'$, we get that the pipeline flow is uniquely determined once the compressor flow is given. If also p_v is given, we can by repeated application of (8) also find p_u for all other nodes $u \in V^v$. Hence FCMP is reduced to finding the flow on all arcs in A_c (Step 2 in Algorithm 1) and p_v for all $v \in V'$ (Step 3).

Step 2 can be approached by identifying cycles in G with negative net cost, as suggested in e.g. [12], and will not be discussed further here. Step 3 can be viewed as follows: Define the *compressor network* (in [12] referred to as the *reduced* network) as the directed graph $G' = (V', A_c')$, where $(u, v) \in A_c'$ if and only if $u, v \in V'$ and there exists some arc in A_c from V^u to V^v. As in [12], we assume that G' does not contain loops, which means that no compressor arc has both it start node and its end node in the same connected component of (V, A_p). The node set of G' can alternatively be associated with the subgraphs G^v, as shown in the illustration of the transition from G to G' (Fig. 1). Optimizing the pressure is now equivalent to solving

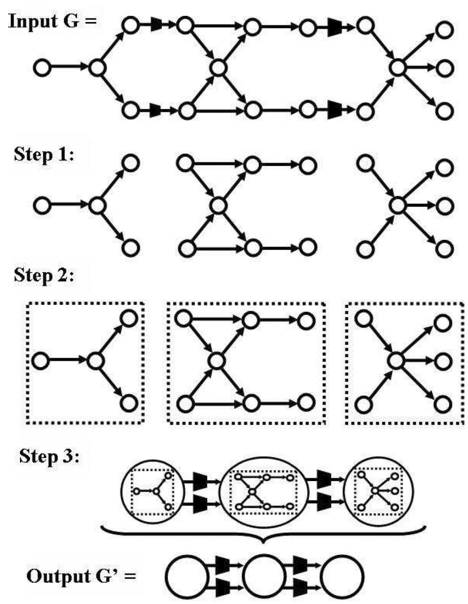

Fig. 1: Transition to compressor network

$$\min_{p \in \Re^{V'}} \left\{ \sum_{(u,v) \in A'_c} g'_{uv}(p_u, p_v) : (p_u, p_v) \in D'_{uv} \forall (u,v) \in A'_c \right\}, \tag{11}$$

where $g'_{uv}(p_u, p_v)$ is the cost incurred on all arcs in A_c between V^u and V^v given that u and v are assigned pressure values p_u and p_v, respectively. Further, D'_{uv} is the feasible domain of (p_u, p_v), taking (7) into account for all arcs from V^u to V^v.

Carter [3] suggested to solve (11) by discretizing $[P^L, P^U]$ and then apply a network reduction technique referred to as *Non-sequential* Dynamic Programming (NDP). Assume that there are m discretization points denoted $p_v^1, \ldots, p_v^m$ for each $v \in V'$, and let $g_{uv}^{ij} = g'_{uv}\left(p_u^i, p_v^j\right)$ if $(p_u, p_v) \in D'_{uv}$ and $g_{uv}^{ij} = \infty$, otherwise. Then NDP consists of a sequence of reductions of G' until the resulting graph is a single node. Three reduction types (see Fig. 2) are considered:

(a) Serial: If $v \in V'$ has exactly two incident arcs (u, v) and (v, t) in G', then v, (u, v) and (v, t) are replaced by a new arc (u, t), and $g_{ut}^{ij} = \min_k \left\{g_{uv}^{ik} + g_{vt}^{kj} : k = 1, \ldots, m\right\}$. The same principle applies if both arcs incident to v enter (leave) v.

(b) Dangling: If $v \in V'$ has only one incident arc (v, t), then t and (v, t) are removed, and, for all in-neighbors u of v in G', g_{uv}^{ij} is updated to $g_{uv}^{ij} + \min_k \left\{g_{vt}^{jk} : k = 1, \ldots, m\right\}$. Similar updates apply to the out-neighbors of v, and the principle applies also if the sole neighbor of t is an out-neighbor.

(c) Parallel: If $k > 1$ arcs $a_1, \ldots, a_k$ in G' connect nodes u and v, then these are replaced by a single arc (u, v). The associated cost parameters are defined as $g_{uv}^{ij} = \sum_{\ell=1}^{k} g_{a_\ell}^{ij}\ \forall i, j = 1, \ldots, m$.

The serial and parallel reductions constitute the preprocessing procedure suggested by Koster et al. [7].

When neither of the reductions (a)-(c) can be carried out, NDP fails. Fig. 3 shows a simple example where this occurs. To overcome this weakness, we now go on to demonstrate how such instances of (11) can be solved.

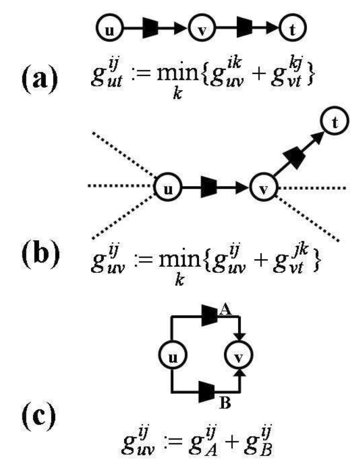

Fig. 2: Network reduction types

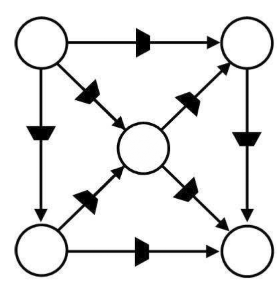

Fig. 3: An instance of G' where NDP fails

4. A TREE DECOMPOSITION APPROACH TO OPTIMIZING THE PRESSURE VALUES

Problem (11) has the mathematical structure of the *frequency assignment problem* [7], and can also be solved by the procedure suggested in [7]. This is based on the following concept introduced by Robertson and Seymour [13]:

Definition 1 *A* tree decomposition *of G' is a pair $\mathcal{J} = (\{X_i : i \in I\}, T)$, where each X_i is a subset of V', called a bag, and T is a tree with node set I. The following properties must be satisfied:*

Algorithm 2 $\mathbf{DP}(\mathcal{J}, i, X, \pi)$

if i is a leaf in T **then**

$$\textbf{return} \min_{p \in \mathcal{D}_{X_i \cup X}} \left\{ \sum_{\substack{(u,v) \in A'_c \\ u,v \in X_i \cup X}} g'_{uv}(p_u, p_v) : p_v = \pi_v \forall v \in X \right\}$$

else

$$\textbf{return} \min_{p \in \mathcal{D}_{X_i \cup X}} \left\{ \sum_{\substack{(u,v) \in A'_c \\ u,v \in X_i \cup X}} g'_{uv}(p_u, p_v) + \sum_{j \in K_i} \mathbf{DP}(\mathcal{J}, j, X_i \cup X, p) : p_v = \pi_v \forall v \in X \right\}$$

- $\bigcup_{i \in I} X_i = V'$;
- *for all* $(u, v) \in A'_c$, *there is an* $i \in I$ *such that* $\{u, v\} \subseteq X_i$;
- $\forall i, j, k \in I$, *if* j *lies on the path between* i *and* k *in* T, *then* $X_i \cap X_k \subseteq X_j$.

The width *of a* tree decomposition $\mathcal{J}$ *is* $\max_{i \in I} |X_i| - 1$.

For any $X \subseteq V'$, define p_X as the vector with components p_v $(v \in X)$ in any consistent order. Define $\mathcal{D}_v = \{p_v^1, \ldots, p_v^m\}$ for all $v \in V'$, and let $\mathcal{D}_X = \{p_X : p_v \in \mathcal{D}_v \forall v \in X\}$. For any $i \in I$, let K_i denote the set of child nodes of i in T.

Algorithm 2 applies dynamic programming to a tree decomposition $\mathcal{J}$ of G'. When bag X_i is to be processed, the union X of all ancestor bags of X_i are input along with a pressure vector $\pi \in \mathcal{D}_X$. The algorithm optimizes the value of p_v for all $v \in X_i$ by complete enumeration of $\mathcal{D}_v$, and by taking into account optimal pressure assignments to all nodes in all child bags of X_i. This is expressed in terms of a recursive call in Algorithm 2. Since $X_i \cap X$ may be nonempty, we must ensure that nodes contained in this set are not assigned new pressure values when processing X_i, and we impose the constraint that $p_v = \pi_v$ for all $v \in X$.

The running time of Algorithm 2 is $\mathcal{O}\left(|I| m^d\right)$, where d is the width of $\mathcal{J}$. This means that finding a tree decomposition of small width can be crucial for the running time of the algorithm. It is however well known [13] that finding one with minimum width is an NP-hard problem, and it is therefore unlikely that a tree decomposition minimizing the running time of Algorithm 2 can be found in polynomial time. We will rely on a heuristic approach to constructing $\mathcal{J}$ with small width.

5. NUMERICAL EXPERIMENTS

To solve (11), we thus apply a two-phase procedure, *TreeDDP*, where the computation of some tree decomposition $\mathcal{J}$ is the first phase, and where Algorithm 2 constitutes the second. The input to this procedure is a network, which is reduced as much as possible by the techniques described in Fig. 2. To compute $\mathcal{J}$, we apply the technique given in [14] based on *Maximum Cardinality Search* [15].

5.1. Test instances

All experiments reported in this work were carried out on the set of test instances shown in Tab. 1. Each row gives an

Tab. 1: Test instances

	Size			$\mathcal{J}$	
Ref	$\lvert V'\rvert$	$\lvert A'_c\rvert$	type	width	$\lvert I\rvert$
A	3	3	4	3	1
B	3	3	5	3	1
C	4	6	1	3	3
D	4	6	2	3	4
E	4	6	3	3	5
F	4	6	4	3	4
G	4	6	5	3	4
H	4	6	6	3	3
I	4	6	7	3	4
J	5	8	4	3	6
K	5	8	8	3	4
L	9	20	4	4	9
M	9	20	5	4	8
N	18	25	2	3	25
O	18	25	4	3	18
P	18	25	9	3	22

identifier of an instance, the size in terms of nodes and arcs in G' after reduction, and the type of compressor used. We consider 9 different compressor types, and all compressors are identical within any given instance. Furthermore, the width and the number of bags in the tree decomposition are given in the two last columns of Tab. 1.

5.2. Experiments

The experiments can be briefly described as follows. The first experiment is a feasibility study where we examine the performance of TreeDDP while varying the granularity of the discretization. We let $m \in \{50, 100, 1000\}$, and let the pressure values be uniformly distributed between their lower and upper bounds.

For a comparison of TreeDDP to a generic global optimization tool, we submit in the second set of experiments (11) to *BARON* [16]. The algorithm of BARON is a variant of branch-and-bound where a convex program is solved in each node of the search tree. We use version 8.1.5 of BARON with version 5.51 of *MINOS* [9] to solve the convex subproblems.

In the third set of experiments, we applied MINOS to compute local optima to problem (11) for 1000 randomly generated starting points.

The TreeDDP procedure was coded in C++ under Linux Red Hat, and all experiments were run on a 2.4 GHz In-

Tab. 2: Performance of TreeDDP

	$m = 50$		$m = 100$		$m = 1000$	
Ref	CPU(s)	Obj	CPU(s)	Obj	CPU(s)	Obj
A	0.0	1.12	0.0	0.77	0.8	0.75
B	0.0	2.63	0.0	2.62	2.2	2.62
C	1.0	10.29	15.9	9.34	245.8	8.79
D	0.1	7.45	11.3	7.34	421.8	7.34
E	1.4	9.66	21.9	6.36	836.1	5.29
F	1.8	6.87	29.5	5.69	1845.2	4.12
G	0.6	9.43	9.5	6.30	1322.5	6.30
H	0.7	6.34	12.7	5.93	712.8	5.09
I	0.6	2.83	9.5	2.82	412.2	2.77
J	0.8	6.07	13.4	5.59	2201.3	5.27
K	0.6	—	9.4	35.67	1052.7	35.67
L	3.1	68.89	49.9	61.83	3424.1	61.73
M	2.5	89.68	39.5	74.80	3092.4	60.74
N	2.1	60.71	34.4	52.46	3554.1	46.00
O	2.6	127.31	41.5	44.51	3623.1	32.62
P	1.4	35.25	23.1	37.67	3417.2	26.54

tel(R) processor with 2 GByte RAM. Experiments with BARON and MINOS were conducted by formulating the model in GAMS [5].

5.3. Results

Table 2 shows the results achieved by TreeDDP while varying m. Instance references are given in the first column, and computation times (CPU-seconds) and objective function values for the respective values of m are given in columns 2-7. The only case where TreeDDP failed to find a feasible solution was for $m = 50$ in instance K. We observe that as m increases, better solutions are found (minimum cost decreases) in all instances, except from a cost increase from $m = 50$ to $m = 100$ in instance P. Nevertheless, a finer discretization also implies, as expected, that the computational requirements increase, and the running time slightly exceeds one CPU-hour in one instance (O).

Table 3 shows the performance of BARON when applied to the test instances. A time limit of 3600 CPU-seconds is imposed, and the relative optimality tolerance is set to 0.01. That is, any feasible solution is considered to be optimal if the gap between the objective function value and its lower bound is below one percent of the objective function value. Columns 2-5 contain the number of iterations in BARON, the maximum number of open nodes the search tree ever had, the objective function value of the best feasible solution found (if any), and the lower bound on the minimum cost.

In 9 out of 16 instances, BARON was able to find a feasible solution, and in 4 instances (A, B, D and J) it was able to prove optimality within the given tolerance. In instances C, E, F, H and L, the relative optimality gap ranged from 2.9% (H) to 50.1% (C), whereas in the remaining instances, no feasible solution was found before the time limit expired. By comparing the last column in Tab. 2 to the lower bounds in Tab. 3, we also observe that the relative optimality gap of TreeDDP in one instance (G) is as large as 64.0%. In the instances where BARON found a feasible solution, the largest gap is 49.4% (instance C).

Tab. 3: Performance of BARON

Ref	Its	#nodes	Obj	LB
A	551	131	0.75	0.75
B	1148	342	2.62	2.62
C	21521	7462	9.02	4.45
D	445	38	7.35	7.28
E	17059	7023	5.30	4.02
F	26765	7480	3.94	2.71
G	5231	1283	—	2.27
H	2109	204	5.19	5.04
I	3267	324	—	2.73
J	27832	2299	5.15	5.10
K	14968	3344	—	20.86
L	740	451	65.94	43.81
M	2438	765	—	31.12
N	1830	839	—	34.28
O	1124	168	—	15.74
P	978	655	—	17.43

Tab. 4: TreeDDP vs. other optimizers

	Minimum cost			TreeDDP vs	
Ref	BARON	MINOS	TreeDDP	BARON	MINOS
A	0.75	0.75	0.75	0.0	0.0
B	2.62	2.62	2.62	0.0	0.0
C	9.02	10.97	8.79	2.5	19.9
D	7.35	7.34	7.34	0.1	0.0
E	5.30	5.63	5.29	0.2	6.0
F	3.94	4.74	4.12	-4.6	13.1
G	—	—	6.30	—	—
H	5.19	5.31	5.09	1.9	4.1
I	—	—	2.77	—	—
J	5.15	5.69	5.27	-2.3	7.4
K	—	—	35.67	—	—
L	65.94	69.16	61.73	6.4	10.7
M	—	61.58	60.74	—	1.4
N	—	—	46.00	—	—
O	—	32.71	32.62	—	0.3
P	—	—	26.54	—	—

In Tab. 4, we compare our results (when $m = 1000$) to the best results obtained by MINOS applied to 1000 randomly generated initial solutions. For an overview, we also include the results from BARON. Columns 2-4 contain the best objective function values obtained by each solver. Whenever applicable, we give in the two last columns the relative cost reduction in percentages when TreeDDP is applied in place of BARON and MINOS, respectively. The numerical values in Column 5 show that neither BARON nor TreeDDP outperforms the other when both are able to compute feasible solutions. We observe that also MINOS failed to find a feasible solution in 5 of the instances, and that it in most instances produced solutions that are of poor quality compared to the TreeDDP and BARON solutions.

6. CONCLUDING REMARKS

In this paper, we have studied a model (FCMP) for minimizing compressor fuel cost in transmission networks for natural gas. An arc in the network model corresponds to either a pipe or a compressor, and the decision variables are arc flow and node pressure. In addition to flow conservation constraints, the model contains non-linear constraints relating pipeline flow to inlet and outlet pressure, as well as non-convex constraints defining the operation domain of the compressors.

Following a general algorithmic idea, which has been suggested and supported experimentally in several recent works, we consider a procedure where each iteration consists of a flow improvement step and a pressure optimization step. Alternating between flow and pressure, one set of decision variables is kept fixed in each step. Still in agreement with previously suggested methods, the non-convex subproblem of optimizing pressure is approximated by a combinatorial one. This is accomplished by discretization of the pressure variables. The contribution of this paper is a method for solving the discrete version of the problem in instances where previously suggested methods fail.

Unlike methods based on successive network reductions, our method does not make any assumptions concerning the sparsity of the network. By constructing a tree decomposition of the network, and apply dynamic programming to it, we are able to solve the discrete version of the pressure optimization problem without enumerating the whole solution space.

We have tested our solution method on a set of imaginary instances, and compared the results to those obtained by applying both a global and a local optimizer to the continuous version of the problem. The experiments indicate that a method guaranteeing the global optimum in reasonable time seems unrealistic even for small instances. Further, discretizing the pressure variables and applying dynamic programming to a tree decomposition gives better results than local optimization, with multiple initial solutions, of the continuous version.

REFERENCES

[1] C. Borraz-Sánchez, R.Z. Ríos-Mercado, "Improving the operation of pipeline systems on cyclic structures by tabu search", *Computers & Chemical Engineering*, Vol. 33, No. 1, pp. 58-64, 2009.

[2] C. Borraz-Sánchez, R.Z. Ríos-Mercado, "A Non-sequential Dynamic Programming Approach for Natural Gas Network Optimization", *WSEAS Transactions on Systems*, Vol. 3, No. 4, pp. 1384-1389, 2004.

[3] R.G. Carter, "Pipeline optimization: Dynamic programming after 30 years", In *Proceedings of the PSIG Meeting*, pp. 1-19, Denver, USA, October 1998.

[4] H.J. Flores-Villarreal, R. Z. Ríos-Mercado, "Efficient operation of natural gas pipeline networks: Computational finding of high quality solutions", In *Proceedings of the International Applied Business Research Conference*, Paper 352, pp. 1-7. Acapulco, Mexico, March 2003.

[5] GAMS Development Corporation, "GAMS: The Solver Manuals. Washington, DC, USA, 2008.

[6] B.J. Gilmour, C.A. Luongo, D.W. Schroeder, "Optimization in natural gas transmission networks: A tool to improve operational efficiency", Technical Report, Stoner Associates Inc., presented at *The Third Meeting on SIAM Optimization*, Boston, USA, April 1989.

[7] A.M.C.A. Koster, S.P.M. van Hoesel, A.W.J. Kolen, "Optimal solutions for frequency assignment problems via tree decomposition", *Lecture Notes in Computer Science*, Vol. 1665, pp. 338-350, 1999.

[8] H.S. Lall, P.B. Percell, "A dynamic programming based gas pipeline optimizer", In A. Bensoussan, J. L. Lions (eds.): *Analysis and Optimization of Systems*, pp. 123-132, Springer, Berlin, Germany, 1990.

[9] B.A. Murtaugh, M.A. Saunders, "MINOS 5.1 User's Guide", Technical Report SOL-83-20R, Systems Optimization Laboratory, Stanford University, Stanford, California, 1983.

[10] A.J. Osiadacz, "Simulation and Analysis of Gas Networks", *Gulf Publishing Company, Houston, USA*, 1987.

[11] R.Z. Ríos-Mercado, S. Kim, E.A. Boyd, "Efficient operation of natural gas transmission systems: A network-based heuristic for cyclic structures", *Computers & Operations Research*, Vol. 33, No. 8, pp. 2323-2351, 2006.

[12] R.Z. Ríos-Mercado, S. Wu, L.R. Scott, E.A. Boyd, "A reduction technique for natural gas transmission network optimization problems", *Annals of Operations Research*, Vol. 117, No. 1-4, pp. 217-234, 2002.

[13] N. Robertson, P.D. Seymour, "Graph minors II. Algorithmic aspects of treewidth", *Journal of Algorithms*, Vol. 7, pp. 309-322, 1986.

[14] S. Subbarayan, "An Empirical Comparison of CSP Decomposition Methods", In: B.Hnich, K.Stergiou (Eds.): *Proceedings of the CP 2007 Doctoral Programme*, pp. 163-168, Providence RI, USA, September 23-27, 2007.

[15] R.E. Tarjan, M. Yannakakis, "Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs", *SIAM Journal of Computing*, Vol. 13, pp. 566-579, 1984.

[16] M. Tawarmalani, N. V. Sahinidis, "Global optimization of mixed-integer nonlinear programs: A theoretical and computational study", *Mathematical Programming*, Vol. 99, No. 3, pp. 563-591, 2004.

[17] P.J. Wong, R.E. Larson, "Optimization of natural gas pipeline systems via dynamic programming", *IEEE Transactions on Automatic Control*, Vol. 13, No. 5, pp. 475-481, 1968.

[18] S. Wu, R.Z. Ríos-Mercado, E.A. Boyd, L.R. Scott, "Model relaxations for the fuel cost minimization of steady-state gas pipeline networks", *Mathematical and Computer Modeling*, Vol. 31, No. 2-3, pp. 197-220, 2000.