Preprints of the 10th IFAC International Symposium on Dynamics and Control of Process Systems
The International Federation of Automatic Control
December 18-20, 2013. Mumbai, India

Branch and Bound Algorithm for Optimal Sensor Network Design

Govind Menon * **M. Nabil** * **Sridharakumar Narasimhan** *

* *Indian Institute of Technology Madras, Chennai 600 036, India*
(e-mail: sridharkrn@iitm.ac.in)

Abstract: The sensor network design procedure developed by Nabil and Narasimhan (2012), which relates process economics and data reconciliation, involves the formulation of a mixed integer cone program. The solution to this problem yields the globally optimal sensor network. A branch and bound method can be used to find the global optimum; however, for systems with large numbers of variables, this approach may require a large amount of computational effort to find the solution. In this paper, a specialized branch and bound algorithm is proposed for solving the sensor network design problem, which uses certain heuristics to obtain a solution faster. One involves a low rank factorization to reduce the size of the relaxed problem. The other involves an approximation of the global lower bound for the branch and bound solution.The utility of this algorithm is demonstrated on a simple flow network, a small but realistic evaporator system, and a medium sized steam metering network.

Keywords: Measurement selection, mixed integer programming, global optimum, data reconciliation, operational loss

1. INTRODUCTION

A process plant requires sensors to aquire good quality data on the variables that describe the process. This data can then be used for several different purposes, including control, monitoring, safety, fault detection, and online optimization. However, a chemical process is usually described by hundreds of variables, and only a limited number of these variables can be measured, owing to the nature of the process and the cost of measurement. The problem of sensor network design for a chemical process typically involves the selection of a few variables to be measured from the complete set of variables that describe the process. The focus of a particular approach to sensor network design can be on different objectives, such as observability of the variables, fault detection and diagnosis (Raghuraj et. al. (1999)), reliability in presence of sensor failure (Ali and Narasimhan (1993)), estimation accuracy (Mah and Kretsovalis (1987)), minimizing capital cost, or maximizing operational profit. Since the individual objectives are usually incommensurable, several approaches have also been proposed, which combine two or more of these objectives (Bhushan and Rengaswamy (2000), Bagajewicz (2002)).

Many of these approaches lead to a graph theoretic formulation of the problem. For example, Raghuraj et. al. (1999) use a problem formulation based on directed graphs. Graph algorithms are then used to solve these problems. Other approaches use an optimization formulation, for example, Bagajewicz (1997) formulates the problem as a mixed integer non-linear program (MINLP).

With the advent of computationally efficient algorithms and tools, the sensor network design procedure of Nabil and Narasimhan (2012) follows an optimization based approach. It combines the objectives of minimum operational loss, and maximum estimation accuracy by data reconciliation. This approach uses an average loss function, defined as a weighted sum of the variances and covariances of individual measurements, to quantify the loss of operational profit. A weighting matrix quantifies the economic importance of each variable. The problem is then formulated as a mixed integer cone program (MICP), which is solved to obtain the globally optimal sensor network. The integer variables in this problem are Boolean.

The resulting optimization problem can be solved using the branch and bound strategy, a widely used approach for solving mixed integer linear programs and mixed integer non-linear programs. Branch and bound methods either guarantee global optimality of the solution, or terminate at a point which can be proved to be ϵ-suboptimal. These methods compute upper and lower bounds on the objective function at every step. The lower bounds, in the case of minimization, are computed by solving the original optimization problem after relaxing some or all of the integer constraints on the variables. Each such relaxed problem forms a 'node' in the branch and bound process, on which further 'branching ' is possible by adding an integer constraint. However, the resulting relaxed problems need not be convex, in which case the algorithm cannot guarantee global optimality of the solution.

The MICP to find the optimal sensor network has been solved to global optimality using the branch and bound solver provided by the YALMIP software package. However, it has been shown that this requires a large amount of computational effort in the case of large networks.

The focus of this work is to develop a branch and bound algorithm specifically for the MICP formulated by Nabil

and Narasimhan (2012). The main objective of developing a new algorithm is to bring about a significant decrease in the amount of computational effort required for sensor network design of large systems using this procedure. There are two features of this algorithm which contribute to attaining this objective.

First, a low rank factorization of the weighting matrix in the loss function is used when solving the relaxed problem at each node. This significantly reduces the size of the resulting semi-definite program and speeds up the overall solution process.

Second, a heuristic is proposed for choosing the current lower bound on the objective function at any stage of the algorithm. This lower bound is then treated as a global lower bound for all nodes that remain to be explored, until a relaxed solution with a smaller function value is encountered. This heuristic, along with the proposed branching strategy, is aimed at reducing the number of nodes visited by the algorithm, while still finding a solution that is close enough to the global optimum.

This paper is organized as follows. In Section 2, the sensor network design procedure and the formulation of the MICP are discussed in brief. Section 3 starts with a brief overview of the general branch and bound strategy. This is followed by a detailed description of the proposed algorithm. Finally, in Section 4, the results of numerical experiments using the proposed algorithm are presented.

2. SENSOR NETWORK DESIGN PROBLEM

This section contains a brief overview of the sensor network design approach developed by Nabil and Narasimhan (2012). This procedure relates process economics and the accuracy of estimates obtained by data reconciliation. For complete details of the sensor network design formulation, the reader is referred to the original article.

To address the data reconciliation objective, this formulation classifies the process variables z as primary variables z_p (any subset of z that forms a minimum observable set) and secondary variables z_s (the remaining variables). With such a segregation of variables, the process model may be described by the equation

$$A_p z_p + A_s z_s = 0 \tag{1}$$

Since the primary variables form a minimum observable network, the matrix A_s is invertible, and the model can be expressed as

$$z_s - B z_p = 0 \tag{2}$$

where $B = -{A_s}^{-1} A_p$. This can be written in the form

$$z = C z_p \tag{3}$$

where z is the vector of all variables of interest, $z = [z_p; z_s]$, and $C = [I; B]$. Here, C is a matrix that relates all the process variables to the primary variables (measurement model). The measurements y can now be expressed as

$$y = C z_p + v \tag{4}$$

where the vector y contains all variables of interest in the process. The data reconciliation problem is formulated as a weighted least squares problem for minimizing error variance in presence of measurement errors, given by

$$\min_{z_p} (y - C z_p)^T Q (y - C z_p) \tag{5}$$

where Q is a weighting matrix of the form

$$Q = \text{diag}\{\frac{q_i}{\sigma_i^2}\} \tag{6}$$

Here q_i is a binary variable indicating whether the i^{th} variable is measured ($q_i = 1$) or not ($q_i = 0$). σ_i^2 is the variance of the measurement error in the i^{th} variable.

This problem is analytically solvable and yields the following error covariance matrix for the variables z:

$$\Sigma_z = C(C^T Q C)^{-1} C^T \tag{7}$$

To address the objective of minimizing the average operational loss, a second order approximation of the cost function is used. The manipulated variables are denoted by u and the disturbance variables are denoted by d. The cost function, $J(u, d)$, describes the negative operational profit of the process accounting for product value, raw material cost, and utility cost. After converting all the variables to the deviation form, the function J is expanded about a nominal operating point (assumed optimal), up to second order terms.

$$J = \frac{1}{2} u^T J_{uu} u + u^T J_{ud} d + \frac{1}{2} d^T J_{dd} d + u^T J_u + d^T J_d. \tag{8}$$

Under the assumption that the only source of uncertainty is the measurement error, this leads to an expression for the average loss L:

$$L = \frac{1}{2} \text{Tr}(W \Sigma_z) \tag{9}$$

Here, W is a positive semidefinite weighting matrix. It depends on the matrices J_{uu} and J_{ud}, which are the second order partial derivatives of the cost function $J(u, d)$. Note that the weighting matrix W must be $(n \times n)$, while the cost function J depends only on the manipulated and disturbance variables, thus giving W the following form:

$$W = \left[\begin{array}{cc|c} \overbrace{J_{ud}^T (J_{uu}^{-1})^T J_{ud}}^{n_d \text{x} n_d} & \overbrace{J_{ud}^T}^{n_d \text{x} n_u} & \overbrace{\mathbf{0}}^{(n_d+n_u)\text{x}n_x} \\ J_{ud} & \underbrace{J_{uu}}_{n_u \text{x} n_u} & \vdots \\ \hline \mathbf{0} & \cdots & \underbrace{\mathbf{0}}_{n_x \text{x} n_x} \end{array}\right]. \tag{10}$$

The resulting expression for average loss is given by

$$L = \frac{1}{2} \text{Tr}(W C (C^T Q C)^{-1} C^T) \tag{11}$$

where Σ_z has been substituted from (7). Using a factorization of the weighting matrix $W = RR^T$, and the Schur complement (see Nabil and Narasimhan (2012)), the problem of minimizing average loss can be formulated as the mixed integer cone program:

$$\min_{t, q_i, Y} = \frac{1}{2} t; \; s.t. \; \text{Tr}(Y) \leq t; \; Y \succ 0$$

$$\left[\begin{array}{cc} Y & R^T C \\ (R^T C)^T & (C^T Q C) \end{array}\right] \succ 0 \tag{12}$$

$$\sum_{i=1}^{n_z} c_i q_i \leq c^*;\ q_i \in \{0,1\};\ Q = \text{diag}\{\frac{q_i}{\sigma_i^2}\}.$$

A full rank factorization of W is obtained using singular value decomposition, such that R is a square matrix of the same size as W. Additionally, in the formulation given in (12), a capital cost constraint is imposed on the selected sensors, where c^* is the available resource limit and c_i is the cost of an individual sensor. This leaves us with the problem of finding the binary variables q_i that minimize the average loss.

As the problem is a mixed Boolean cone program, it can be solved using a branch and bound procedure. The constraints in optimization problem (12) are linear in Q. Therefore, the problems resulting from the relaxation of the integer constraints on the diagonal elements of Q at each node, will be convex. The relaxations can thus be solved to global optimality by a semi-definite programming solver. This means that the branch and bound method can potentially find a solution whose global optimality is guaranteed.

Globally optimal solutions to the above problem have been found for test cases, using the branch and bound solver available in the YALMIP software package (J. Löfberg (2004)). This solution procedure was found to be computationally demanding in the case of even moderately sized systems, often requiring several hours of computational time to find the optimum. The reason for this is twofold. The first is the direct increase in number of integer variables. The second is the increase in size of the matrix Y. Clearly, Y is a square matrix of the same size as R, which, in turn, is the same size as W $(n_z \times n_z)$. The entries of Y are decision variables of the optimization problem, which means that any increase in the size of Y translates to a much larger relaxed problem.

Thus, when designing a sensor network for larger systems, the size of W becomes larger, and the corresponding increase in size of Y increases the amount of computational effort required to solve the relaxed problem. For this approach to be effectively used in designing sensor networks for very large systems therefore, we need a more computationally efficient way of finding the optimum.

3. ALGORITHM

In this section, an overview of the general branch and bound approach is provided, followed by a detailed description of the proposed algorithm for the solution of the MICP resulting from the sensor network design procedure.

3.1 General Branch and Bound Strategy

The branch and bound solution approach can be applied to problems of several different types. It is a popular method for solving mixed integer programming problems, both linear and non-linear. This approach works on the principle that the set of all solutions can be partitioned into smaller subsets, which can then be evaluated separately, following a systematic procedure, until a solution is found that can be proved to be globally optimal. In the case of mixed integer programming, the branch and bound strategy needs to be used in conjunction with a solution approach for the relaxed problem. In our case, the relaxed problems are solved by semi-definite programming.

If the branch and bound algorithm is not terminated prematurely, it is guaranteed to find an optimal solution to linear and convex nonlinear problems. However, in most cases, in the interest of reducing computation time, the algorithm is terminated at a stage when the incumbent integer solution can be proved to be ϵ-suboptimal.

Consider a general mixed integer optimization problem of the form

$$\begin{array}{ll} \underset{x,z}{\text{minimize}} & f_0(x,z) \\ \text{subject to} & f_i(x,z) \leq b_i,\ i = 1,\ldots,m. \\ & z_j \in \mathbb{Z},\ j = 1,\ldots,n. \end{array} \tag{13}$$

Here, the variables x are continuous variables, while the variables z are restricted to integer values. The functions f may be linear or nonlinear. In the case of mixed Boolean problems, as in our case, the constraints on z would be $z_j \in \{1,0\}$.

The branch and bound algorithm begins by relaxing all the integer constraints on the variables. The solution to this relaxed problem need not have all the z variables at integer values. The algorithm then *branches* on one of the z variables which is at a non-integer value. The branching step involves the addition of an integer constraint on the chosen variable. A new subproblem is formed every time the algorithm branches.

At each new subproblem, *bounds* are computed, on the optimum objective function value. The lower bound is calculated by solving the problem with the integer constraints relaxed on all the remaining variables. The global upper bound at any stage of the algorithm is the smallest objective function value (for minimization) corresponding to an integer solution.

At the outset, the upper bound may be calculated in different ways, such as simply rounding each of the relaxed Boolean variables z_j to 0 or 1, or first rounding each of the relaxed Boolean variables to 0 or 1, and then, with these values of fixed, solving the resulting convex problem in the variables x (Boyd and Mattingley (2007)).

Different heuristics exist to decide which variable to branch on, and which subproblem to investigate next. These heuristics use conditions on the relaxed variables and the lower bound of the subproblem to decide the next step.

At any stage of the algorithm, a global upper bound and lower bound are defined. In some cases, when the exact global optimum is not needed, the algorithm is terminated when the gap between these two bounds becomes smaller than a given tolerance ϵ. Otherwise, the algorithm terminates only when all nodes are completely fathomed, i.e., when it is certain that branching further on any of the remaining non-integer nodes will not lead to a better solution, thus guaranteeing global optimality.

In most cases, the condition that all nodes are fathomed can only be satisfied after a large amount of computational time. Therefore, it becomes preferable to terminate the algorithm at a point which is ϵ-suboptimal. For example, in the current problem, a tolerance of 0.01 would mean

that the optimum sensor network found gives an average loss that is within one percent of the loss given by the globally optimum network.

3.2 Proposed Algorithm

The development of a specialized branch and bound solution procedure for sensor network design, using the approach of Nabil and Narasimhan (2012), has been motivated in Section 2. The heavy computational demands for large systems, at present, limits the applicability of the method to systems with few variables.

The focus of the proposed branch and bound approach is therefore to minimize the computational effort required in finding optimum solutions to large systems, thereby facilitating the easy application of this method to systems of any size.

The main features of the proposed algorithm are discussed below.

Solving the relaxed problem: As mentioned before, the relaxation of the integer constraints at any subproblem result in a convex optimization problem, for which a solution can be found that is guaranteed to be globally optimal. In the present formulation (given in (12)), the relaxation is a semi-definite programming problem, for which several solvers are available.

As a way of reducing the size of the relaxed problem, we propose the use of a low rank factorization of the weighting matrix W. Since the cost function J usually depends only on a small number of process variables, W is typically a highly sparse matrix. This makes a low rank factorization highly effective in reducing the size of the relaxed problem. The factorization is computed by first performing a singular value decomposition on W, as before. However, we now neglect those singular values which account for less than 0.1 percent of the sum of singular values. Thus we have

$$\begin{aligned} W &= U_1 S_1 U_1^T + U_2 S_2 U_2^T \\ &\approx U_1 S_1 U_1^T. \end{aligned} \tag{14}$$

where U_1 and S_1 correspond to the retained singular values. Thus $R = U_1 S_1^{\frac{1}{2}}$. If p singular values are retained, U_1 will be an $n_z \times p$ matrix and S_1 will be a non-singular $p \times p$ matrix, which would make R an $n_z \times p$ matrix. Thus, the matrix Y in the relaxed problem would become $p \times p$, significantly reducing the computational effort when p is much less than n_z.

Branching strategy: In common branch and bound solvers, the variable selection for branching is based on some criterion, such as picking that variable whose value is closest to one or zero, picking the one whose value is closest to 0.5, or using information about the Lagrange multipliers corresponding to each constraint (Boyd and Mattingley (2007)). The branching strategy used in the present approach is simpler, in that it does not use any heuristic for deciding the branching variable. It simply selects the first non-integer variable q_i when the variables are arranged in order from $i = 1, \ldots, n$.

Another simplification in the branching strategy lies in the selection of the next subproblem to be investigated. Instead of selecting that subproblem with the smallest lower bound, this algorithm selects the nearest neighbor to the subproblem that was investigated in the previous step. The nearest neighbor is defined in terms of the binary tree structure implicit in this solution approach.

Pruning strategy: The method used here for discarding a given subproblem, i.e., pruning, is the same as that used in most branch and bound approaches. A subproblem is pruned if its lower bound exceeds the current global upper bound.

Global upper bound: The global upper bound at any stage of the branch and bound solution process is the lowest objective function value corresponding to an integer solution. The upper bound value is used at every step to discard those subproblems which are certain to be suboptimal. The lower the value of the upper bound, the greater the chance that a given subproblem whose branches do not lead to a better integer solution, will be discarded.

The algorithm starts with the completely relaxed problem, and branches on the first non-integer variable q_i (see the optimization problem described by (5)). This variable is set to 0 and 1, and the corresponding relaxed subproblems are solved. According to the proposed method, to find a good upper bound, the next branching will be on that subproblem that gives the smaller optimum value between the two. This adds two new subproblems, from which the one with the smaller function value is chosen for branching.

This heuristic is followed until an integer solution is found, meaning that more integer constraints will continue to be added until an integer solution is reached. The optimum at this solution now becomes the global upper bound for the branch and bound algorithm. It remains the global upper bound until a This approach is commonly used in branch and bound solvers.

Global lower bound: The lower bound at a particular subproblem is found by solving the corresponding relaxed optimization problem, which in this case, is a semi-definite program. The global lower bound at any stage of the branch and bound is the objective function value corresponding to that subproblem that has the smallest lower bound among those that are yet to be explored.

The branching strategy used in the present approach, as explained previously, is different from that of the traditional branch and bound, and the function values of all the remaining unexplored subproblems will not be available at an intermediate stage.

The proposed heuristic works in tandem with the branching strategy used. The global lower bound is approximated to be the minimum of the available lower bounds, over all the fathomed subproblems at the current *level*. Here, the *level* is defined as the number of integer constraints which are active at a given subproblem.

For example, suppose the algorithm is currently about to explore the nodes resulting from a parent node, at which three of the variables are constrained to integer values. The *level* of the parent node here is three. The current lower bound is set as the minimum of the objective function

values over all the subproblems of *level* three that have been evaluated by the algorithm until the current step.

Note that this may not give us the global lower bound over all the remaining unexplored nodes, which a traditional branch and bound approach would use. The lower bound used here is thus an approximation to the actual global lower bound.

Termination: There are three ways in which the proposed algorithm can terminate. The first, is if the original problem, with all the integer constraints relaxed, is found to be infeasible. The second, is if all possible subproblems have been either pruned or completely fathomed. The third condition for termination uses a tolerance ϵ, for the gap between the current global upper bound and lower bound. The gap is defined as given by Edgar et. al. (2001):

$$\text{gap} = \frac{|\textit{upper bound} - \textit{lower bound}|}{1 + |\textit{upper bound}|}$$

If the gap is less than the tolerance ϵ, the algorithm terminates. In all the numerical experiments described in the next section, a value of 0.01 is used as the tolerance.

4. NUMERICAL EXPERIMENTS

In this section, we present the results of applying the proposed branch and bound technique to the sensor network design problem for three different systems. The first, a small system, is the flow network of an ammonia process described in Narasimhan and Jordache (2000). The second, a slightly larger system, is a realistic evaporator system described in Kariwala et. al. (2008). The third is a steam metering network described in Narasimhan and Jordache (2000), which is an example of a moderately sized system, with 28 variables.

4.1 Algorithm Implementation

The algorithm was implemented using `Matlab R2010a`, on a Windows system, with an Intel Core 2 Duo 2.20 GHz and 4 GB of RAM. The semi-definite programming solver available with the `cvx` convex optimization package was used for solving the relaxed subproblems.

4.2 Ammonia Network

The sensor network design problem for the simplified ammonia process was solved by Nabil and Narasimhan (2012), as a test case for their approach. The manipulated variables are F_5 and F_7, while F_1 is considered as the disturbance variable. For $z_p = [F_2\ F_5\ F_7]^T$, the process matrix is as given by Nabil and Narasimhan (2012). A cost function of the form $J = (F_5 - F_7)^2 + (F_5 - F_1)^2$ is used to find the weighting matrix W. It is assumed that the costs of the individual sensors are same and the capital cost is available for selecting only the minimum number of sensors.

Results: The resulting MICP was solved using the proposed branch and bound algorithm. The optimum sensor network was found to be $\{F_1, F_5, F_8\}$. The optimum solution was compared with that obtained using the YALMIP branch and bound solver, and found to match. The proposed algorithm visits 7 nodes, while the YALMIP solver visits 8. For this small system, the there is almost no difference in effort between the two solvers.

4.3 Evaporator System

The optimal sensor network design approach was also applied to a realistic evaporation process by Nabil and Narasimhan (2012). The forced-circulation evaporator system is shown in Fig. 1. Here, the concentration of the feed stream is increased by evaporating the solvent through a vertical heat exchanger with circulated liquor.

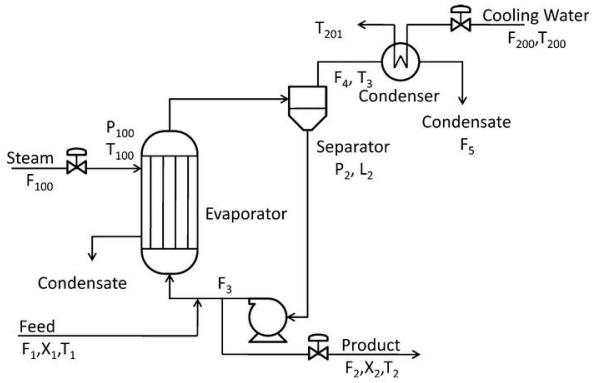

Fig. 1. Evaporator system.

The economic objective is to minimize the loss of operational profit. A cost function of the form

$$J = 600F_{100} + 0.6F_{200} + 1.009(F_2 + F_3) + 0.2F_1 - 4800F_2 \quad (15)$$

is used. The nominal values are obtained by solving the resulting nonlinear optimization problem (Kariwala et. al. (2008)).

The variables $d = [X_1\ T_1\ T_{200}]^T$ are taken as disturbance variables, with $X_1 = 5\%$, $T_1 = 40°C$, and $T_{200} = 25°C$ at the nominal operating point. The manipulated variables are $u = [F_{200}\ F_1]^T$, and the remaining variables of interest are $x = [F_2\ F_3\ F_4\ F_5\ F_{100}\ T_2\ T_3\ T_{201}\ P_2\ Q_{100}\ Q_{200}]^T$. The vector of all variables of interest is therefore $z = [d;\ u;\ x]$. The primary variables are chosen to be $z_p = [F_2\ F_3\ F_{100}\ F_{200}\ T_{201}]^T$. The implementation, or measurement errors for flow and pressure measurements are taken to be 2% and 2.5% of the nominal values, respectively. The available capital cost is taken to be \$550. The process matrix is obtained by linearization around the operating point. The weighting matrix is obtained numerically using the objective function (15).

Results: The resulting MICP was solved using the proposed branch and bound algorithm. The optimum sensor network was found to be $\{F_2, F_3, F_{100}, F_{200}, T_{201}\}$. The optimum solution was compared with that obtained using the YALMIP branch and bound solver, and found to match. The proposed algorithm visits 8 nodes, while the YALMIP solver visits 16. For this system also, the there is very little difference in efficiency between the two solvers.

4.4 Steam Metering Network

To evaluate the computational efficiency of the proposed algorithm, we have implemented it on a larger problem.

For this purpose, we use the steam metering system of a methanol synthesis plant, as described by Narasimhan and Jordache (2000). This system contains 11 process units and 28 process streams. This is a moderately sized problem with 28 variables, with a minimum of 17 sensors required.

The cost function was randomly generated such that J_{uu} is positive definite ($J_{uu} \succ 0$) and well-conditioned, and the resulting MICP was solved for 50 different cases, using both the proposed algorithm and the branch and bound solver provided by YALMIP. The gap tolerance was set to 0.01 for both solvers. For a particular case, the problem was also solved using the proposed algorithm, without the low rank factorization, for comparison. This could not be repeated for all 50 cases because of the large amount of computational time required by the YALMIP solver.

Table 1. Numerical Experiments

	Ammonia network	Evaporator system	Synthetic network
Number of variables	8	16	28
Number of primary variables	3	5	17
Number of possible combinations	56	4368	21474180
Number of nodes explored by proposed algorithm	7	8	1065*
Number of nodes explored by YALMIP	14	16	22115*
Computational time using proposed algorithm with low rank factorization (s)	7.61	9.80	1047.22*
Computational time using YALMIP with low rank factorization (s)	-	-	3287.15*
Optimum average loss ($/h) using proposed algorithm	3	10.28	32.03*
Optimum average loss ($/h) using YALMIP	3	10.28	30.74*

*Average over 50 runs.

Results: For the 50 randomly generated cases of the steam metering network problem, the average number of nodes visited by the proposed algorithm was found to be lower than the number visited by the YALMIP solver, by a factor of approximately 22. However, the optimum value of the loss found by the proposed algorithm was found to be, on average, about 4 percent higher than that found by the YALMIP solver. This additional loss results from the approximation of the lower bound used by the algorithm.

For the single case where the proposed algorithm was used with and without the low rank factorization, the number of nodes visited remained the same in both cases, but the computational time was found to increase by approximately a factor of 16. This result clearly showcases the effectiveness of the low rank factorization in improving computational efficiency.

5. CONCLUSION

A branch and bound type algorithm, combined with a low rank factorization, has been proposed for solving the sensor network design problem of Nabil and Narasimhan (2012). From the results presented for the steam metering network, we can conclude that the proposed algorithm is successful in reducing the computational effort required to solve the sensor network design problem. However, the solution found by the proposed algorithm is found to have, on average, a loss that exceeds the global optimum by about 4 percent, while the YALMIP solver guarantees a loss within 1 percent of the global optimum. Since the aim has been to make the sensor network design procedure of Nabil and Narasimhan (2012) applicable to systems much larger than the those dicussed in the paper, the additional loss, which results from the approximation of the lower bound used by the algorithm, may be considered an acceptable penalty to be paid for the significant reduction in computational effort. Work is in progress to demonstrate the utility of the approach for very large systems, and to identify other heuristics that can improve the algorithm.

ACKNOWLEDGEMENT

This work was partially supported by the Board of Research in Nuclear Sciences under the project "Experiment design on convex optimization."

REFERENCES

M. Nabil and S. Narasimhan. Sensor network design for optimal process operation based on data reconciliation. *Ind. Eng. Chem. Res.*, 51:6789–6797, 2012.

R. Raghuraj, M. Bhushan, and R. Rengaswamy Locating sensors in complex chemical plants based on fault diagnostic observability criteria. *AIChE J.*, 45:310–322, 1999.

Y. Ali and S. Narasimhan Sensor network design for maximizing reliability of linear processes. *AIChE J.*, 39:820–828, 1999.

M. Bhushan and R. Rengaswamy. Design of sensor location based on various fault diagnostic observability and reliability criteria. *Comput. Chem. Eng.*, 24:734–741, 2000.

R. S. Mah and A. Kretsovalis. Effect of redundancy on estimation accuracy in process data reconciliation. *Chem. Eng. Sci.*, 42:2115–2121, 1987.

M. Bagajewicz. A review of techniques for instrumentation design and upgrade in process plants. *Can. J. Chem. Eng.*, 80:3–16, 2002.

M. Bagajewicz. Design and retrofit of sensor networks in process plants. *AIChE J.*, 43:2300–2306, 1997.

S. Boyd and J. Mattingley. Branch and bound methods. Notes for EE364b, Stanford University, 2007.

S. Narasimhan and C. Jordache. *Data reconciliation and gross error detection: An intelligent use of process data* Gulf Publishing Co.: Houston, TX, 2000.

V. Kariwala, Y. Cao, and S. Janardhanan. Local self-optimizing control with average loss minimization. *Ind. Eng. Chem. Res.*, 47:1150–1158, 2008.

T. F. Edgar, D. M. Himmelblau, and L. S. Lasdon. *Optimization of Chemical Processes.* McGraw-Hill Co.: New York, NY, 2001.

J. Löfberg . YALMIP : A Toolbox for Modeling and Optimization in MATLAB. *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.