I&EC research
Industrial & Engineering Chemistry Research

Article

pubs.acs.org/IECR

Efficient Approximate Methods for the Design and Upgrade of Sensor Networks

DuyQuang Nguyen and Miguel J. Bagajewicz*

University of Oklahoma, 100 E. Boyd Street, T-335 Norman, Oklahoma 73019

ABSTRACT: A rigorous method based on process equations for designing sensor networks for nonlinear systems was recently proposed (Nguyen and Bagajewicz, Design of Nonlinear Sensor Networks for Process Plants. *Ind. Eng. Chem. Res.* **2008**, *47*, 5529−5542). The method has been shown to be efficient for middle scale problems. For realistic large scale problems, the method can identify several suboptimal solutions but fails to identify the optimal solution within an acceptable PC computational time. This paper presents an approximate method that attempts to locate optimal solution starting from the suboptimal solutions provided by the rigorous equations-based method. We show that although optimality is not guaranteed, the proposed method was able to identify global optimum solutions in design case studies.

1. INTRODUCTION

Improved process monitoring via data reconciliation and appropriate gross error detection is achieved and/or improved by proper systematic location of sensors in a process plant. The sensors are located to achieve the desired thresholds for key variables, on various attributes like precision, error detectability, resilience, etc. by means of data treatment techniques based on process material and component and energy balances. The problem of an optimum selection of sensor location is referred to as sensor network design.

Research on sensor network design is extensive. A review up to the year 2000 can be found in the book by Bagajewicz.[1] The most common objective function is minimum cost, and the desired properties of a sensor network (the specifications) can be categorized into three types: (i) desired process monitoring capabilities with requirements on observability, redundancy, precision threshold values, and gross error detectability, etc. of key variables, (ii) desired process fault detection capabilities with specifications on process fault resolution and reliability of fault monitoring system, (iii) although still intended for process monitoring purposes, desired reliability of estimability of key variables rather than the degree/precision of the estimation of key variables itself. Variants to the common formulation do exist, for example, maximum precision or maximum reliability at budget constraints. Most of the researches focus on type (i), that is, designing a sensor network for process monitoring purposes. Only a few articles deal with type (ii) (Raghuraj et al.;[2] Bhushan and Rengaswamy[3−5]) using graph algorithms to design a sensor network for detecting and identifying process faults. In addition, Narasimhan and Rengaswamy[6] used an economic objective, reflecting the savings obtained from being able to detect the faults and the costs of the instrumentation. Bagajewicz et al.[7] designed a sensor network for simultaneous process monitoring and fault detection/resolution using a MILP formulation. Most recently, Bhushan et al.[8] presented a framework for designing a robust sensor network for reliable process fault diagnosis, which was then solved by using constraint programming (Kotecha et al.,[9,10]). Finally, Chen and Chang[11] used a graph-theoretic method to design a sensor network for process fault identification.

Ali and Narasimhan[12] were the first to introduce the concept of network reliability, which relates to the probability of estimability of key variables in the presence of sensor failures. On the basis of this concept, they used graph-theoretic methods to design a sensor network maximizing network reliability at a limited number of sensors for linear, redundant linear, and bilinear systems, respectively (Ali and Narasimhan[12−14]). Finally, Bagajewicz and Sanchez[15] presented a cost optimal formulation based on reliability targets. These are the only works devoted exclusively for reliability-based sensor network design. Reliability has been used in the multicriterion design framework of Sen et al.[16] and Carnero et al.[17]

There are many papers on sensor network design for process monitoring purposes; a few of them are described here with emphasis on computational methods. Kretsovalis and Mah[18] minimized a weighted sum of estimation error and measurement cost using a combinatorial search algorithm. Madron and Veverka[19] used Gauss Jordan elimination to achieve observability of all key variables at minimum sensor cost. Chmielewski et al.[20] showed that an unmeasured variable can be modeled in data reconciliation formulation using a fake sensor with very high variance; this technique was later used in many papers. They used linear matrix inequalities (LMI) technique to solve a relaxed problem in each node of the branch and bound method, which guarantees global optimum. Heyen et al.[21] used a genetic algorithm to design a cost-optimal sensor network; the computation algorithm is applicable to nonlinear systems by linearization of process constraints at the nominal operating conditions, assuming steady state. Kelly and Zyngier[22] presented a MILP model based on the Schur complements

Special Issue: L. T. Fan Festschrift

Received: September 24, 2011
Revised: February 29, 2012
Accepted: April 11, 2012
Published: April 11, 2012

theorem to design a sensor network for process monitoring purposes.

Recently, the multiobjective sensor network design became attractive. Bagajewicz and Cabrera[23] addressed multiobjective sensor network design using the pareto optimal solutions visualization techniques. Sen et al.[16] and Carnero et al.[17,24] used genetic algorithms. Finally, Bagajewicz and Sanchez[25] showed that there is a duality of some sort in which minimum cost models subject to reliability constraints are related to maximum reliability models subject to cost constraints.

Mathematical programming techniques were used by Bagajewicz and co-workers. Bagajewicz[26] was the first to formulate the problem as a mixed-integer programming problem by using binary variables to indicate whether a variable/stream is measured or not; the resulting MINLP model was solved by a branch and bound method. Bagajewicz and Cabrera[27] reformulated the problem as a MILP model, which was solved in the commercial GAMS software package. In addition, different new sensor network attributes were also introduced (software accuracy, stochastic accuracy, etc.[28]). Departing from the traditional cost-optimal approach, Nguyen and Bagajewicz[29] presented a model that maximizes economic value of information minus the cost of the sensor network.

More recently, the computational efficiency of the branch and bound method presented in Bagajewicz[26] was remarkably improved by using cutsets (instead of individual measurements) and a decomposition technique (Gala and Bagajewicz[30,31]). Most recently, Gala and Bagajewicz's cutsets-based methods, which are applicable only to linear systems, were extended to nonlinear systems by using equations in the tree search instead of cutsets (Nguyen and Bagajewicz[32]). The method is rigorous (i.e., it guarantees optimality) but its computational performance is not efficient enough for solving realistic large-scale nonlinear problems.

This work addresses the shortcoming of our equation-based method by using a heuristic local search attempting to locate optimal solution starting from suboptimal solutions provided by equation-based method. The paper is organized as follows: first an overview of the sensor network design problem is presented, followed by a brief description of the equation-based method; the proposed approximate method is then presented. Finally, two illustrated examples are provided.

2. SENSOR NETWORK DESIGN PROBLEM

The popular cost-optimal sensor network for process monitoring purpose is formulated as follows (Bagajewicz[26]):

$$\left.\begin{aligned} &\min \sum_{\forall i} c_i q_i \\ &\text{s.t.} \\ &\sigma_i(q) \le \sigma_i^* \quad \forall\, i \in M_S \\ &q_i = 0, 1 \quad \forall\, i \end{aligned}\right\} \tag{1}$$

where q_i is a binary variable indicating that a sensor is located in variable/stream i, c_i is the cost of sensor i, M_s represents the set of variables where a certain specification is required (desired level of precision/residual precision or error delectability, etc.), $\sigma_i(q)$ is the value of the property under consideration (e.g., precision, accuracy, etc), and σ_i^* is the corresponding threshold.

While the inequalities referred to in the above model seem intuitively correct, they are not simple to obtain, much less write explicitly in a closed form equation. Rather, they may require a procedure to be implemented. For example, precision can be obtained analytically once the position and the precision of each sensor is known, but when the position of each sensor is described through a binary variable, a closed formula includes products of binaries and continuous variables and involves the inverse of a matrix (see Bagajewicz and Cabrera[27]). When accuracy is sought, then the procedure involves solving a mathematical optimization problem for software accuracy or the use of a Monte Carlo simulation for stochastic accuracy (see Bagajewicz[28]). This is the reason why our group has resorted to tree enumeration and others to genetic algorithms,[21] while there is no literature addressing the solution of this problem using mathematical programming methods that include accuracy (only a few attempts using LMI-based models[20] or MILP models [27] were presented for just precision constraints). Finally, in the above model, there are threshold values (σ_i^*), which are difficult to determine a-priori and which lead to multiobjective formulations. Recent work, has resorted to the use of an alternative model that maximizes the economic value of information minus cost,[29] making the choice of thresholds for accuracy, precision, and reliabilities to be made automatically because they are tied to the economics of the problem.

Two main groups of computational methods have been used to solve the above problem: integer programming methods (for example, integer programming method was used to solve the MILP model described in Bagajewicz and Cabrera[27]), these methods guarantee optimality but they usually exhibit scaling problem, and stochastic methods (e.g., genetic algorithms, which do not guarantee optimality). Our group uses tree search (an integer programming method) and exploits certain properties that allow fathoming branches (a brief summary can be found in Nguyen and Bagajewicz[32]). The equation-based tree search is briefly reviewed here because it is used as a preprocessing step for the approximate method presented in this work.

3. EQUATION-BASED TREE SEARCH METHOD

This method is briefly described as follows (more details can be found in Nguyen and Bagajewicz[32]). It is based on exploring a tree of nodes, where each node corresponds to all the sensors that measure key variables in the corresponding equation.

1. Find all the equations of the process
2. Keep only the equations containing key variables (we call them candidate equations).
3. Sort these candidate equations in ascending order of their cost (the cost of an equation is equal to sum of the costs of the sensors used to measure variables contained in that equation).
4. Start with the root node with no equation being added, that is, **e** = {0, 0, 0, ...), trivially infeasible.
5. Using the branch first rule, develop each branch by making one element of **e** active and adding one candidate equation at a time which is chosen from the remaining equations using a branching criterion.
6. While implementing the branching criteria, if any set of equations has already been evaluated in previous nodes, that node is not developed any further, that is, its children nodes are ignored. This occurs frequently because one set of measurements can be a result of the union of different combinations of equations.

dx.doi.org/10.1021/ie202186y | Ind. Eng. Chem. Res. 2013, 52, 83−90

7. This is continued until the stopping criterion is met. In such case, the algorithm backs up two levels and develops the next branch.

Branching Criterion. While exploring the tree from one node to the other, either going down the tree or exploring the sister node, the newly added equation is chosen in such a way that the cost obtained by its union with the existing active equations is minimal.

Stopping Criterion. Because adding an equation always increases the cost, whenever a feasible node is found (one that satisfies all the constraints of the problem), the tree is not explored further down.

The method was further improved by using a decomposition technique. The idea is to decompose the original system (represented by incidence matrix) into subsystems so as to reduce the number of equations in the candidate lists, hence reduce the size of the tree. Because the system is decomposed, there are some missing equations in the candidate list when compared with the list of equations of the original system, which are equations containing variables from different subsytems. Fortunately, these missing equations can be found while exploring down the tree using a Gaussian elimination operation on equations. The tree search procedure is almost the same as the procedure without decomposition except that (i) the branching and stopping criterion are modified and (ii) in each node, Gaussian elimination operations between active equations are performed to find the aforementioned missing equations.

The equation-based tree search method coupled with decomposition has been shown to be efficient for medium scale nonlinear problems. For realistic large scale problems, it quickly identifies several good (suboptimal) solutions but fails to locate an optimal solution within an acceptable time. More details on the merits and the limitations of the equation-based methods can be found in Nguyen and Bagajewicz.[32]

4. APPROXIMATE METHOD

We now propose to improve the equation-based method by incorporating a local search after some part of the tree has been developed. The idea is to use the good solutions provided by the equation-based method as input in a local tree search procedure to hopefully arrive at global optimum (the term "good" solution is meant to be a feasible solution with objective value/cost near to that of the global optimum). The procedure is as follows: (1) Run the equation-based tree search (branch and bound) method, record the current best solutions identified by the tree search method. Terminate the equation-based tree search after a predefined number of nodes are explored or computational time has been reached. These current best solutions are the aforementioned "good solutions" that are provided by the equation-based method. (2) Run the two-step heuristic local search using the obtained good solutions as input to identify a better solution, which is highly likely to be the true optimal solution.

The proposed local search method was developed on the basis of the following observation: the global optimal solution and near-optimal solutions belong to the same region in the space of variables; that is, they are different from one another in values of only a few variables (in this context, the measurement locations). This is due to the inherent characteristics of the sensor network: if a measurement is good because the associated sensor is cheap and this measurement contributes significantly to the observability and redundancy of key variables, then it will very likely show up in global optimum and some other "good" solutions. Compared to the global optimum, a good solution usually misses one or two good measurements and contains some other "extra" measurements. Therefore, it is reasonable to assume that all the measurements showing up in good solutions are good measurements that are very likely to show up in global optimum

On the basis of the above arguments we present the heuristic local search as the following two-step procedure.

Step One. The purpose of this step is to find a low cost feasible solution which will contain a number of those "good" measurements. After the regular tree search has been conducted up to a certain point, the calculation procedure to find such a solution is as follows: (1) Find the union of the last five current best solutions provided by the equation-based method (find all good measurements that show up in the last five current best solutions), denoted as vector **U**; (2) Employ a tree enumerative strategy to remove measurements out of vector **U** to obtain a minimum cost solution (the method is essentially the same as the inverted tree search described in Nguyen and Bagajewicz[32]). The minimum cost solution is denoted as **MC.**

If all good measurements belonging to the global optimum actually show up in the last five current best solutions (which is highly probable), the identified minimum cost solution **MC** is indeed a global optimum. However, there is a very small chance that one or two good measurements belonging to the global optimum do not show up in vector **U**, thus we go on to step two to account for such a situation.

Step Two. The purpose of this step is to determine if it is possible to improve the solution by replacing a certain number of measurements in **MC** by some other measurements not belonging to **MC**. This is done by exploring all the possibilities of replacing a certain number of measurements (denoted as *Nr*, a parameter) in **MC** with elements in vector **A** (the set of measurements that are not in **MC**) trying to obtain a solution better than **MC**. The procedure is as follows: (1) Remove a certain set of *Nr* measurements out of **MC** and denote the resulting vector as **B**; (2) Use a tree enumerative strategy to add elements (measurements) from **A** into **B** trying to obtain feasible solutions with minimum cost. More specifically, the tree search algorithm starts from **B** as root node, then continues adding measurements contained in **A** into the root node (vector **B**) until a feasible solution is found, then steps back one level and explores a new branch. When all branches have been explored, the tree search is terminated; (3) Remove another set of *Nr* measurements out of **MC** thus obtaining the new vector **B** and repeat the same procedure; (4) Terminate the process when all the possibilities of removing *Nr* measurements out of **MC** are explored.

To remove *Nr* measurements out of **MC**, all possible combinations of *Nr* measurements are explored. If the number of elements in vector **MC** is n_t, then the number of such combinations is $C_{Nr}^{nt} = n_t!/(N_r!(n_t - N_r)!)$. The value of *Nr* is a parameter (can be any value, typically smaller than $n_t/2$). The greater is the value of *Nr*, the greater is the value of C_{Nr}^{nt}, hence the longer is the computational time. Testing results suggest that the best values for *Nr* are 1 and 2 (explanations are provided in the examples section). The two-step procedure is illustrated in Figure 1.

All the proposed methods and the approximate method described in this paper as well as the other branch and bound methods used to validate the obtained solutions are

dx.doi.org/10.1021/ie202186y | *Ind. Eng. Chem. Res.* 2013, 52, 83−90

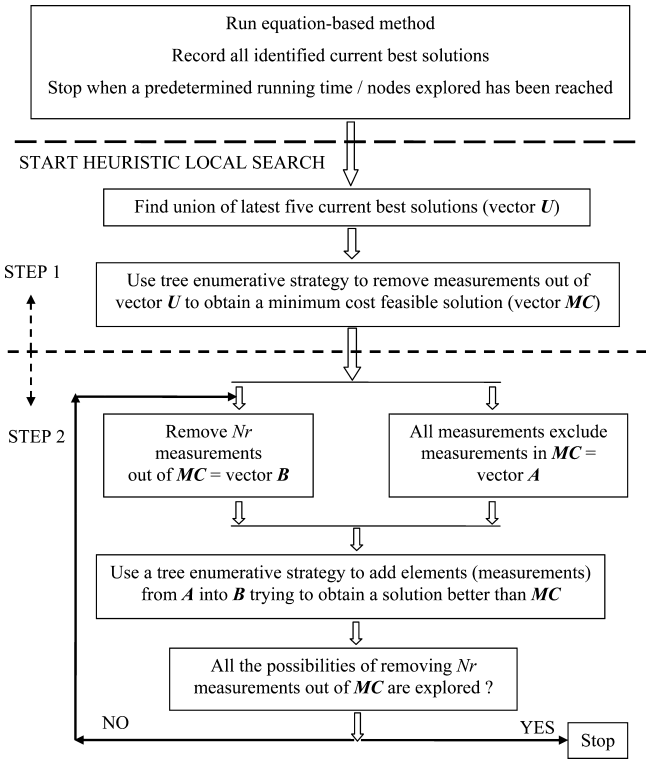

Figure 1. Algorithm flowchart for the approximate method

implemented in Fortran running on a 2.8 GHz Intel Pentium 1028 MB RAM PC.

5. EXAMPLES

5.1. Mineral Flotation Process Example. Consider a middle scale process, the mineral flotation process (MFP) introduced by Smith and Ichiyen,[33] shown in Figure 2.

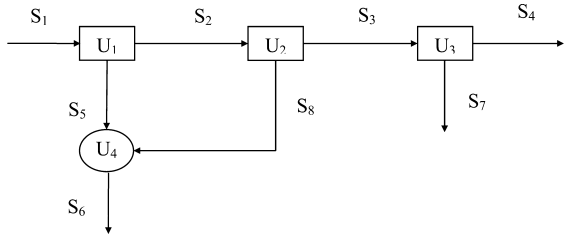

Figure 2. The mineral flotation process.

The process consists of three flotation cells (separators) and a mixer. Each stream consists of two minerals, copper and zinc, in addition to gangue material. The total flowrate F and the composition of copper C_A and zinc C_B are variables of interest, so the total number of variables under consideration is 24. Let us assume that each variable can be measured separately by a sensor (there may be situation that one analyzer can measure both copper and zinc composition, in that case we can lump the two composition variables of copper and zinc into one single variable). The process model is as follows:

Total flowrate balance equation around one unit (U_j, j = 1, 4):

$$\sum_i F_i = 0$$

Copper component flowrate balance equation around one unit (U_j, j = 1, 4):

$$\sum_i F_i C_{Ai} = 0$$

Zinc component flowrate balance equation around one unit (U_j, j = 1, 4):

$$\sum_i F_i C_{Bi} = 0$$

The total number of original balance equations is 12. The component balance equations are nonlinear, hence the system is nonlinear (it is a bilinear system). The nominal operation condition is given in Table 1 (taken from Narasimhan and Jordache[34]) and the nonlinear constraints are linearized using a Taylor expansion around the nominal operation condition. All sensor precisions are 2%. The sensor costs are given in Table 2.

Table 2. Sensor Costs for Mineral Flotation Process Example (×10 USD)

streams:	1	2	3	4	5	6	7	8
F_i	50	55	45	60	40	48	52	58
C_{Ai}	300	310	240	260	250	360	320	335
C_{Bi}	290	350	330	340	280	270	295	275

Two design case studies are considered. They are described in Table 3.

Table 3. Design Case Studies for the Mineral Flotation Process Example

	MFP1	MFP2
no. of key variables	5	10
key variables	F_1, F_4, F_6, C_{B1}, C_{A6}	F_1, F_4, F_6, F_7, C_{A1}, C_{B1}, C_{B4}, C_{A6}, C_{B7}, C_{A8}
requirement	observability	observability
precision thresholds	2%	2%
measured variables	F_3, F_4, F_6, C_{B1}, C_{A6}	F_4, F_5, F_6, F_7, C_{A1}, C_{B1}, C_{B4}, C_{A5}, C_{B7}, C_{A8}
sensors cost	803	2010

Two methods were used to solve the problem: (i) the rigorous tree search method (based on individual measurements, described in Bagajewicz[26]); this method guarantees optimality; (ii) the approximate method. For the approximate method, the calculation proceeds until step 1 only (the

Table 1. Nominal Operation Condition for the Mineral Flotation Process

streams:	1	2	3	4	5	6	7	8
F_i (kmol/hr)	100	92.67	91.57	84.48	7.33	8.43	7.09	1.1
C_{Ai} (% mol)	0.019	0.0045	0.0013	0.001	0.2027	0.2116	0.0051	0.2713
C_{Bi} (% mol)	0.0456	0.0437	0.0442	0.0041	0.069	0.0495	0.5227	0.001

dx.doi.org/10.1021/ie202186y | Ind. Eng. Chem. Res. 2013, 52, 83−90

calculation is terminated when vector **MC** has been obtained). This is done to investigate the performance (the ability to identify global optimal solution) of the approximate method if only step 1 is used.

Both methods identify the same solutions, which are global optimal solutions. The solutions are described in row 5 (measured variables) and row 6 (sensors cost) of the Table 3.

The computational times of the two methods are described in Table 4. When solving the design case MFP2, the equation-based tree search is terminated after 50 000 nodes have been explored, then the heuristic local search is applied (it explores further 180 nodes) and identifies the optimal solution.

Table 4. Performance of the Approximate Method, the MFP Example

case study		MFP1	MFP2
rigorous tree search	number of nodes explored	25 293	3 230 514
	computational time	1 min	1 h, 51 min
approximate method	number of nodes explored	2 246	50 180
	computational time	12 s	9 min 48 s

It can be found from the above results that the approximate method is much more efficient than the rigorous tree search method and that, although optimality is not guaranteed, the approximate method identifies the optimal solution.

5.2. The Tennessee Eastman (TE) Process Example. The TE process flowsheet is given in Figure 3. The same process model and data as given in Nguyen and Bagajewicz[32] are used. The readers are referred to literature source on TE process such as Ricker and Lee[35] and Nguyen and Bagajewicz[32] for detailed description of the model. The nominal values of variables and the sensor costs are given in Table 5.

Values of flowrates F_i are given in kmol/h; P_r and P_s are the pressures in reactor and separator, respectively (KPa); T_r and T_s are the temperatures in reactor and separator, respectively (K); subscripts *A, B, C, D, E, F, G, H* denote components; subscripts 6, 7, 8, 9, 10, and 11 denote stream number. The variables listed in Table 6 are considered as candidates for measurements, other variables in the TE process (e.g., input flowrates F_1, F_2, and F_3) are assumed to be either known by measurements (forced measurements) or of little importance

Table 5. Data for the Tennessee Eastman Problem

variables	nominal operating condition	sensor cost	variables	nominal operating condition	sensor cost
F_6	1889.9	300	$Y_{E,8}$	0.186	740
F_7	1475.2	300	$Y_{F,8}$	0.023	730
F_{10}	258.56	200	$Y_{G,8}$	0.048	740
F_{11}	211.3	200	$Y_{H,8}$	0.023	750
$Y_{A,6}$	0.322	770	$Y_{A,9}$	0.33	720
$Y_{B,6}$	0.089	780	$Y_{B,9}$	0.138	730
$Y_{C,6}$	0.264	730	$Y_{C,9}$	0.24	740
$Y_{D,6}$	0.069	740	$Y_{D,9}$	0.013	750
$Y_{E,6}$	0.187	750	$Y_{E,9}$	0.186	760
$Y_{F,6}$	0.016	760	$Y_{F,9}$	0.023	770
$Y_{G,6}$	0.035	810	$Y_{G,9}$	0.048	780
$Y_{H,6}$	0.017	820	$Y_{H,9}$	0.023	790
$Y_{A,7}$	0.272	750	$Y_{D,10}$	0.002	700
$Y_{B,7}$	0.114	760	$Y_{E,10}$	0.136	710
$Y_{C,7}$	0.198	700	$Y_{F,10}$	0.016	720
$Y_{D,7}$	0.011	710	$Y_{G,10}$	0.472	720
$Y_{E,7}$	0.177	720	$Y_{H,10}$	0.373	730
$Y_{F,7}$	0.022	730	$Y_{G,11}$	0.537	730
$Y_{G,7}$	0.123	780	$Y_{H,11}$	0.438	740
$Y_{H,7}$	0.084	790	P_r	2806	100
$Y_{A,8}$	0.33	780	T_r	393.6	500
$Y_{B,8}$	0.138	770	P_s	2734.7	100
$Y_{C,8}$	0.24	760	T_s	353.3	500
$Y_{D,8}$	0.013	750			

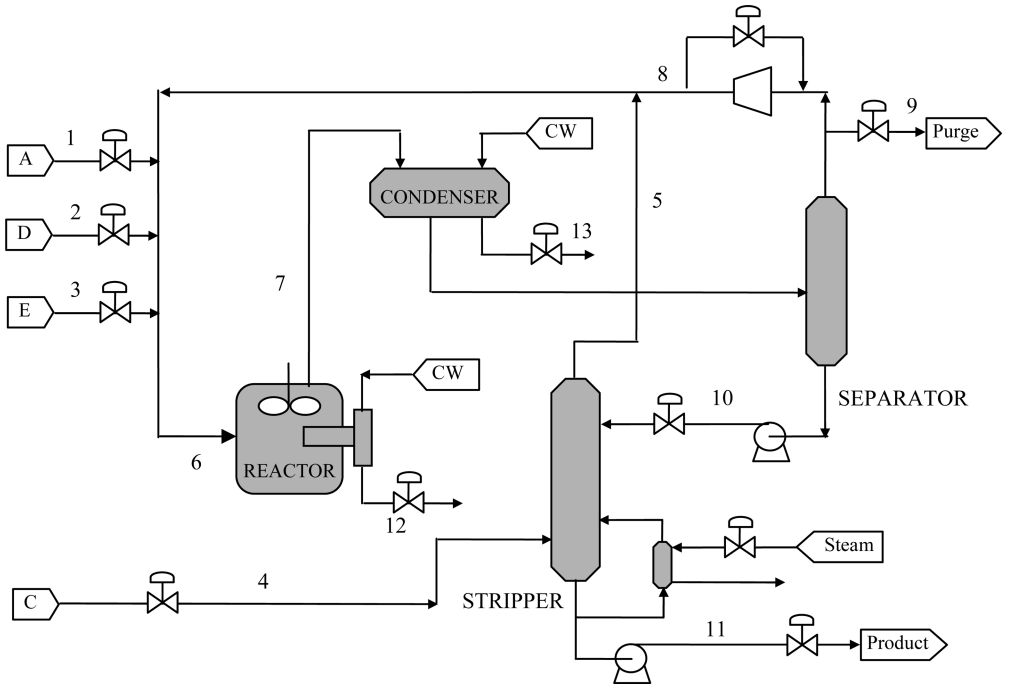

Figure 3. The Tennessee Eastman Process (following Downs and Vogel[37]).

dx.doi.org/10.1021/ie202186y | *Ind. Eng. Chem. Res.* 2013, 52, 83−90

Table 6. Design Case Studies for the TE Process Example

design case	TE1 moderate spec.	TE2 high spec.	TE3 high spec.
no. of key variables	17	19	23
key variables	F_6, y_{A6}, y_{G6}, y_{H6}, F_7, y_{G7}, y_{H7}, y_{A9}, y_{G9}, y_{H9}, F_{11}, y_{G11}, y_{H11}, P_r, T_r, P_s, T_s	F_6, y_{A6}, y_{B6}, y_{G6}, y_{H6}, F_7, y_{A7}, y_{B7}, y_{C7}, y_{A8}, y_{B8}, y_{C8}, y_{D8}, y_{A9}, y_{B9}, y_{C9}, F_{11}, y_{G11}, y_{H11}	F_6, y_{A6}, y_{B6}, y_{G6}, y_{H6}, F_7, y_{A7}, y_{B7}, y_{C7}, y_{G7}, y_{H7}, y_{A8}, y_{B8}, y_{C8}, y_{D8}, y_{A9}, y_{B9}, y_{C9}, y_{G10}, y_{H10}, F_{11}, y_{G11}, y_{H11}
requirement	observability	redundancy	redundancy
precision thresholds	2%	1.5%	1.5%
residual precision thresholds		4%	4%
measured variables	F_6, y_{A6}, y_{G6}, y_{H6}, F_7, y_{A7}, y_{A9}, y_{G9}, y_{H9}, F_{11}, y_{G11}, P_s (12 sensors)	All variables but $\{y_{C6}, y_{E6}, y_{F6}, y_{E9}, y_{F9}, y_{E10}, y_{F10}, y_{G10}, y_{H10}, P_r, T_r, P_s, T_s\}$ (34 sensors)	All variables but $\{y_{C6}, y_{E6}, y_{F6}, y_{E9}, y_{F9}, F_{10}, y_{E10}, y_{F10}, P_r, T_r, P_s, T_s\}$ (35 sensors)
sensors cost	7070	23 560	24 810

for consideration. The total number of equations involving listed variables is 28.

Three design case studies are considered, which are described in Table 6. The first design case is the one with moderate specification described in Nguyen and Bagajewicz.[32] The other two design cases are ones with a high level of specifications.

Except for design case studies with low level of specifications (where feasible solutions contain only a small fraction of available candidate sensors), the TE example cannot be solved by using the rigorous individual measurement-based tree search (described in Bagajewicz[26]). It is estimated that, if moderate or high levels of specification are required, solving the TE example by individual measurement-based tree search takes as long as several weeks. The equation-based tree search method coupled with decomposition (called Decomposed Equations in Nguyen and Bagajewicz[32]) is the only viable option for design cases with a moderate level of specifications while design cases with a high level of specifications can be solved by using either the inverted tree strategy (Nguyen and Bagajewicz[32]) or the Decomposed Equations method.

We use Decomposed Equations to obtain several good solutions as starting point, then use local search to arrive at optimum solution. The calculation procedure follows exactly the procedure described in Figure 1 with the following values for the two parameters: terminating the equation-based method after 100 000 nodes has been explored and the parameter *Nr* is 2.

For the two design cases TE2 and TE3 (high specification), the inverted tree search method described in and Bagajewicz[32] was used to validate the solutions obtained by the proposed approximate method (both methods solve the two design cases TE2 and TE3 within a few hours but the inverted tree search guarantees optimality). For the first design case TE1, its solution was validated by using the level by level "L by L" tree search described in Nguyen and Bagajewicz.[36] The "L by L" tree search method also guarantees optimality (it takes roughly four days for the "L by L" method to solve the design case TE1; if the inverted tree search is used to solve the design case TE1, the computational time is estimated to be several weeks). Thus, the solutions obtained by the approximate method are validated (by the inverted tree search method and the "L by L" tree search method described in our previous papers) to be optimal solutions. Note that for all three design case studies, the optimal solution is already found in local search step 1.

The computational performance of the approximate method is shown in Table 7.

Although the approximate method does not guarantee optimality, in all the testing problems we have tried, the combination of Decomposed Equations method and local search step 1 is able to locate optimal solution. The local search step 2 is a safeguard step to avoid the possibility of missing optimal solution. Let us denote the measurements contained in the optimal solution as optimal measurements (in the opposite side, the rest are called nonoptimal measurements). The optimal solution is missed only if the following two situations occur *simultaneously*: (i) the mentioned combination fails to locate optimal solution, that is, the current best **MC** is not the optimal solution (which means that **MC** contains some nonoptimal measurements) and (ii) the number of nonoptimal measurements in **MC** is more than the number of measurements we consider removing out of **MC** (that is, the parameter *Nr*).

Table 7. Performance of the Approximate Method, TE Process Example

		TE1 moderate spec.	TE2 high spec.	TE3 high spec.
number of nodes explored	local search step 1	48 544	484	212
	local search step 2	54 097	130 683	166 244
	total	202 641	231 167	266 456
total computational time		1 h 12 min	1 h 33 min	1 h 40 min

Thus, practically speaking, it is reasonable to *assume* that the solution obtained by the approximate method is a true global optimal solution for engineering applications (where computational time is as important as optimality). Thus, the computational efficiency and the capability to locate an optimal solution (although this fact is not theoretically proven) of the approximate method make it the most suitable method for solving realistic large scale nonlinear sensor network design problems.

The chosen value of the *Nr* parameter has an affect in two aspects. (1) Optimality: increasing the value of *Nr* parameter increases the chance of finding a global optimal solution when the local search step 1 misses the optimal solution. (2) Computational time: increasing the value of *Nr* parameter profoundly increases the computational time.

Thus, there is a tradeoff in selecting the value for the *Nr* parameter. However, because the local search step 1 is usually able to identify the global optimal solution, the step 2 is just a safeguard step (as proven in the two shown examples); thus, it is then reasonable to focus only on the affect of the *Nr* parameter on computational time, which suggests using a small

dx.doi.org/10.1021/ie202186y | *Ind. Eng. Chem. Res.* 2013, 52, 83−90

value for *Nr* to reduce computational time. We did some tests on the dependence of computational time on the parameter *Nr* (the TE process example was used for the tests) and found that computational time for $Nr \geq 4$ is too long (in a magnitude of a few days, this figure is problem-specific). While computational time for $Nr = 3$ is acceptable (several hours for design case TE1 to almost a day for design cases TE2 and TE3), it is recommended to use $Nr = 1$ or 2.

Table 7 shows that the proposed method is able to efficiently solve a large scale nonlinear problem, the TE example. It found optimal solutions in less than 2 h. More details on the method are given as follows:

For design case studies TE2 and TE3: after exploring 100 000 nodes, the Decomposed Equations identifies four current best solutions that contain 39, 37, 38, and 36 sensors, respectively (costs are 26990, 26290, 26240 and 25540). The union of these solutions (vector **U**) contains 39 sensors, which is exactly the same as the first current best identified (this means that the first current best contains all "good" measurements). Exploring all the possibilities of removing sensors out of vector **U** results in the optimal solutions (containing 34 and 35 sensors with costs being 23,560 and 24,810 for design cases TE2 and TE3, respectively).

For design case TE1: after exploring 100 000 nodes, the Decomposed Equations method identifies 11 current best solutions; the last five solutions have costs ranging from 11840 to 13370 and the number of sensors ranging from 20 to 22. The union of the last five solutions (vector **U**) contains 23 sensors. Using enumerative tree search strategy to remove the sensors out of **U** (exploring 48 544 nodes) results in the optimal solution that contains only 12 sensors costing 7070. This solution is much better than the current best solution obtained by using the Decomposed Equations method only (that solution, described in Nguyen and Bagajewicz,[32] contains 17 sensors whose cost is 9630).

6. CONCLUSIONS

In this work, an approximate method is proposed to solve nonlinear sensor network design problems that rigorous tree search methods are incapable of solving in reasonable time. The approximate method is very efficient; it is able to solve large scale nonlinear problems within a couple of hours. Although it does not guarantee optimality, the chance of finding a global optimal solution is very high. Indeed, the proposed method was able to find optimal solutions in all the design case studies shown in this paper.

AUTHOR INFORMATION

Corresponding Author

*Tel.: (405) 325-5458. Fax: (405) 325-5813. E-mail: bagajewicz@ou.edu.

Notes

The authors declare no competing financial interest.

REFERENCES

(1) Bagajewicz, M. *Design and Upgrade of Process Plant Instrumentation*; Technomic Publishers: Lancaster, PA, 2000.

(2) Raghuraj, R.; Bhushan, M.; Rengaswamy, R. Locating Sensors in Complex Chemical Plants Based on Fault Diagnostic Observability Criteria. *AIChE J.* **1999**, *45* (2), 310−322.

(3) Bhushan, M.; Rengaswamy, R. Design of Sensor Network Based on the Signed Directed Graph of the Process for Efficient Fault Diagnosis. *Ind. Eng. Chem. Res.* **2000**, *39*, 999−1019.

(4) Bhushan, M.; Rengaswamy, R. Comprehensive Design of Sensor Networks for Chemical Plants Based on Various Diagnosability and Reliabilty Criteria. I. Framework. *Ind. Eng. Chem. Res.* **2002**, *41*, 1826−1839.

(5) Bhushan, M.; Rengaswamy, R. Comprehensive Design of Sensor Networks for Chemical Plants Based on Various Diagnosability and Reliabilty Criteria. II. Applications. *Ind. Eng. Chem. Res.* **2002**, *41*, 1840−1860.

(6) Narasimhan, S; Rengaswamy, R. Quantification of Performance of Sensor Networks for Fault Diagnosis. *AIChE J.* **2007**, *53* (4), 902−917.

(7) Bagajewicz, M.; Fuxman, A.; Uribe, A. Instrumentation Network Design and Upgrade for Process Monitoring and Fault Detection. *AIChE J.* **2004**, *50* (8), 1870−1880.

(8) Bhushan, M.; Narasimhan, S.; Rengaswamy, R. Robust Sensor Network Design for Fault Diagnosis. *Comput. Chem. Eng.* **2008**, *32*, 1067−1084.

(9) Kotecha, P. R.; Bhushan, M.; Gudi, R. D. Constraint Programming Based Robust Sensor Network Design. *Ind. Eng. Chem. Res.* **2007**, *46*, 5985−5999.

(10) Kotecha, P. R.; Bhushan, M.; Gudi, R. D. Design of Robust, Reliable Sensor Networks Using Constraint Programming. *Comput. Chem. Eng.* **2008**, *32*, 2030−2049.

(11) Chen, J. Y.; Chang, C. T. Development of an Optimal Sensor Placement Procedure Based on Fault Evolution Sequences. *Ind. Eng. Chem. Res.* **2008**, *47*, 7335−7346.

(12) Ali, Y.; Narasimhan, S. Sensor Network Design for Maximizing Reliability of Linear Processes. *AIChE J.* **1993**, *39* (5), 820−828.

(13) Ali, Y.; Narasimhan, S. Redundant sensor network design for linear processes. *AIChE J.* **1995**, *41* (10), 2237−2249.

(14) Ali, Y.; Narasimhan, S. Sensor Network Design for Maximizing Reliability of Bilinear Processes. *AIChE J.* **1996**, *42* (9), 2563−2575.

(15) Bagajewicz, M.; Sanchez, M. Cost-Optimal Design of Reliable Sensor Networks. *Comput. Chem. Eng.* **2000**, *23*, 1757−1762.

(16) Sen, S.; Narasimhan, S.; Deb, K. Sensor Network Design of Linear Processes Using Genetic Algorithms. *Comput. Chem. Eng.* **1998**, *22*, 385−390.

(17) Carnero, M.; Hernandez, J.; Sanchez, M.; Bandoni, A. An Evolutionary Approach for the Design of Nonredundant Sensor Networks. *Ind. Eng. Chem. Res.* **2001**, *40*, 5578−5584.

(18) Kretsovalis, A.; Mah, R. S. H. Effect of Redundancy on Estimation Accuracy in Process Data Reconciliation. *Chem. Eng. Sci.* **1987**, *42*, 2115.

(19) Madron, F.; Veverka, V. Optimal Selection of Measuring Points in Complex Plants by Linear Models. *AIChE J.* **1992**, *38* (2), 227.

(20) Chmielewski, D.; Palmer, T.; Manousiouthakis, V. On the Theory of Optimal Sensor Placement. *AIChE J.* **2002**, *48* (5), 1001−1012.

(21) Heyen, G.; Dumont, M.; Kalitventzeff, B. Computer-aided design of redundant sensor networks. In *Proceeding of 12th European Symposium on Computer-aided Process Engineering*; Grievink, J., van Schijndel, J., Eds.; Elsevier Science: Amsterdam, 2002, 685−690.

(22) Kelly, J. D.; Zyngier, D. A New and Improved MILP Formulation to Optimize Observability, Redundancy and Precision for Sensor Network Problems. *AIChE J.* **2008**, *54* (5), 1282−1291.

(23) Bagajewicz, M.; Cabrera, E. Pareto Optimal Solutions Visualization Techniques for Multiobjective Design and Upgrade of Instrumentation Networks. *Ind. Eng. Chem. Res.* **2003**, *42*, 5195−5203.

(24) Carnero, M.; Hernandez, J.; Sanchez, M.; Bandoni, A. On the Solution of the Instrumentation Selection Problem. *Ind. Eng. Chem. Res.* **2005**, *44*, 358−367.

(25) Bagajewicz, M.; Sánchez, M. Duality of Sensor Network Design Models for Parameter Estimation. *AIChE J.* **1999**, *45*, 661−664.

(26) Bagajewicz, M. Design and Retrofit of Sensors Networks in Process Plants. *AIChE J.* **1997**, *43* (9), 2300−2306.

(27) Bagajewicz, M.; Cabrera, E. A New MILP Formulation for Instrumentation Network Design and Upgrade. *AIChE J.* **2002**, *48*, 2271−2282.

(28) Bagajewicz, M. *Smart Process Plants: Software and Hardware for Accurate Data and Profitable Operations*. (ISBN:978−0-07−160471), McGraw Hill: New York, 2010.
(29) Nguyen, D.; Bagajewicz, M. New Sensor Network Design and Retrofit Method Based on Value of Information. *AIChE J.* **2011**, *57* (8), 2136−2148.
(30) Gala, M.; Bagajewicz, M. Rigorous Methodology for the Design and Upgrade of Sensor Networks Using Cutsets. *Ind. Eng. Chem. Res.* **2006**, *45* (20), 6687−6697.
(31) Gala, M.; Bagajewicz, M. Efficient Procedure for the Design and Upgrade of Sensor Networks Using Cutsets and Rigorous Decomposition. *Ind. Eng. Chem. Res.* **2006**, *45* (20), 6679−6686.
(32) Nguyen, D.; Bagajewicz, M. Design of Nonlinear Sensor Networks for Process Plants. *Ind. Eng. Chem. Res.* **2008**, *47* (15), 5529−5542.
(33) Smith, H. W.; Ichiyen, N. Computer Adjustment of Metallurgical Balances. *Can. Znst. Mining Metall. (C.Z.M.) Bull.* **1973**, *66*, 97−100.
(34) Narasimhan, S.; Jordache, C. *Data Reconciliation and Gross Error Detection: An Intelligent Use of Process Data*; Gulf Publishing: Houston, TX, 2000.
(35) Ricker, N. L.; Lee, J. H. Nonlinear Modeling and State Estimation for the Tennessee Eastman Challenge Process. *Comput. Chem. Eng.* **1995**, *19* (9), 983−1005.
(36) Nguyen, D.; Bagajewicz, M. New Efficient Breadth-First/Level Traversal Tree Search Methods for the Design and Upgrade of Sensor Networks. *AIChE J.* **2011**, *57* (5), 1302−1309.
(37) Downs, J. J.; Vogel, E. F. A Plant-Wide Industrial Process Control Problem. *Comput. Chem. Eng.* **1993**, *17* (3), 245−255.