AIChE

New Sensor Network Design and Retrofit Method Based on Value of Information

DuyQuang Nguyen and Miguel J. Bagajewicz
School of Chemical, Biological and Materials Engineering, The University of Oklahoma, Norman, OK 73019

DOI 10.1002/aic.12440
Published online November 2, 2010 in Wiley Online Library (wileyonlinelibrary.com).

Traditionally, the sensor network design procedure was based on positioning sensors so that certain network monitoring capabilities (e.g., observability, redundancy, and error detectability) of key variables are assured at minimum sensors cost. We present a new approach that is based on maximizing economic value of information minus cost instead of the traditional approach that requires the satisfaction of performance targets. This article presents the conceptual aspect and computation issues of this new approach: the connection between the new approach and the traditional minimum-cost approaches is explored and the computational methods to solve the proposed problem are presented. © 2010 American Institute of Chemical Engineers *AIChE J,* 57: 2136–2148, 2011
Keywords: instrumentation design, sensor network design, optimization methods

Introduction

The problem of optimum selection of sensor location is referred to as sensor network design problem (SNDP). Because of economic reasons, not every process variable of interest can be measured by a sensor. In the context of data treatment techniques like data reconciliation and gross errors detection, the location of measured points (i.e., location of sensors) has direct effect on the accuracy of estimators of variables of interest (key variables), which in turn affects process plant performance because these estimators are used to make decisions.

Prior works on the SNDP are extensive. Bagajewicz[1] offers a description and a review of almost all published works up to year 2000. Despite the large amount of published research works on this problem, the scope/target of the SNDP is surprisingly limited. The SNDP can be divided into two classes.

Class one: designing sensor network for process monitoring purposes

More specifically, the sensor networks are designed to provide accurate estimators (measured or estimated value) for the process variables of interest (key variables). The most popular model formulation is to find cost-optimal sensor network satisfying a certain number of pre-specified requirements (e.g., observability and redundancy of key variables). The problems of data reconciliation and gross error detection of partly measured systems are inherent parts of the process monitoring-focused SNDP. A different approach that also belongs to this class is that of estimation reliability (instead of the estimation precision) where it is considered either as an objective or a constraint (this approach, based on the concept of system reliability, was introduced by Ali and Narasimhan[2]). A few examples of research works on process monitoring-targeted SNDP are: Madron and Veverka[3] used multiple Gauss Jordan elimination to achieve observability of all key variables at minimum sensor cost. Meyer et al.[4] and Luong et al.[5] used graph-theoretic methods. Chmielewski et al.[6] used branch and bound method with linear matrix inequalities transformation to obtain a solution. Sen et al.[7] and Carnero et al.[8,9] used genetic algorithms (GAs). Most recently, Kelly and Zyngier[10] presented a mixed integer linear programming (MILP) model based on the Schur complements theorem to design sensor network for process monitoring purpose. Branch and bound (tree search) methods were particularly used in our research group. Among them the tree search methods based on cutsets (Gala and Bagajewicz[11,12]) were shown to be very efficient

Correspondence concerning this article should be addressed to M. J. Bagajewicz at bagajewicz@ou.edu.

for linear systems. In addition, the equation-based method (Nguyen and Bagajewicz[13]) and level traversal methods (Nguyen and Bagajewicz[14]) were developed for nonlinear systems.

Class two: designing sensor network for process fault detection and identification purposes

This problem is based on the principle that a process fault (malfunction/failure in an instrument in a process) at one point in the system will propagate to other locations in the system, which would eventually be detected by the sensors-based monitoring system. The problem of detecting and identifying faults (using various well-established techniques) is an inherent part of the problem. A few examples of research works on SNDP for process fault diagnosis are described next. Raghuraj et al.[15] and Bhushan and Rengaswamy[16,17,18] presented sensor network design formulation based on fault diagnosis criteria. Musulin et al.[19] used GA in the design of sensor network for principal components analysis monitoring. Bhushan et al.[20] presented a framework for designing robust sensor network for reliable process fault diagnosis; the problem was then solved by using constraint programming (Kotecha et al.[21,22]).

All the published work on SNDP for process monitoring purposes focuses on finding more efficient computational methods to solve the problem in which the sensors' cost is minimized and the popular specifications on precision, residual precision, error detectability and resilience, and estimation reliability are used as performance targets. There is trade-off between cost and performance (process monitoring capability) of the sensor network: if one asks for higher level of performance, one would have to pay more in sensors' cost (use more sensors). When there is such a trade-off, the right strategy is to simultaneously optimize both cost and performance of the sensor network. This simultaneous optimization of performance and cost can be done by solving the design problem using different levels of design specifications, thus constructing a pareto optimal curve (Bagajewicz and Cabrera[23]). The right strategy to do this kind of optimization is obtaining an economic indicator for performance of the network and then maximizing the difference between the economic indicator and cost of the sensor network. This would be the utility function of the multiobjective (cost and performance) optimization. We call this approach a value-optimal SNDP, the traditional minimum sensors cost approach with requirements on performance targets being termed cost-optimal SNDP. This approach has been conceptually discussed in the seminal paper by Bagajewicz et al.[24] From then on, the work by Narasimhan and Rengaswamy[25] is the only published work that discusses the value of a sensor network as a performance measure from the fault diagnosis perspective).

From the monitoring perspective, the following works paved the way to obtain the value of information: Bagajewicz[26] introduced the concept of software accuracy that essentially encompasses all the aforementioned performance measures (observability, redundancy, error detectability, etc.) of sensor network. The economic value of software accuracy was also quantified (Bagajewicz[27]) and an efficient approximate method was developed to evaluate the economic value of accuracy (Nguyen et al.[28]). Thus, the first necessary step in developing the value-optimal SNDP, the quantification of economic indicator of performance of sensor network, has been solved; the problem remaining is to investigate the value-optimal SNDP and to find efficient methods to solve the proposed problems.

This article presents a new approach to design sensor networks (for process monitoring purposes) that maximizes the economic value of accuracy (value-optimal SNDP). Relationship between this new approach and the traditional cost-optimal approach is discussed and computational methods to solve the problem are presented.

This article is organized as follows: first, the concepts of software accuracy and economic value of accuracy are briefly reviewed, followed by description of computational methods to evaluate software accuracy and its associated economic value. The value-optimal SNDP and efficient methods to solve the proposed problems are then presented.

Software Accuracy

Accuracy has been defined as precision plus bias (Miller[29]). However, the definition is of less practical use because the bias size is generally unknown. Recently, Bagajewicz[26] introduced the concept of software accuracy in the context of data reconciliation and gross error detection being used to detect biases. In such context, accuracy was defined as sum of precision and induced bias instead of the actual bias. The induced bias $\hat{\delta}$ and the software accuracy are shown next (Bagajewicz[26]):

$$\hat{\boldsymbol{\delta}} = E[\hat{\boldsymbol{x}}] - \boldsymbol{x} = [\mathbf{I} - \mathbf{SW}]\boldsymbol{\delta} \tag{1}$$

$$\hat{a}_i = \hat{\sigma}_i + \delta_i^* \tag{2}$$

where $\hat{a}_i, \hat{\sigma}_i, \delta_i^*$ are the accuracy, precision (square root of variance S_{ii}) and the induced bias of the estimator, respectively. Also $\mathbf{I}$ is the identity matrix, $\mathbf{S}$ is the variance matrix of measurements, and $\mathbf{W}$ is calculated as $\mathbf{W} = \mathbf{A}^T (\mathbf{ASA}^T)^{-1} \mathbf{A}$ ($\mathbf{A}$ is process constraints matrix), and $\boldsymbol{\delta}$ is the vector of actual biases in measurements.

By definition, the accuracy value relies on how one calculates the induced bias. From Eq. 1, it is clear that the induced bias is the function of undetected biases whose sizes can be any value below the threshold detection values and their location can be anywhere in the system. Thus, the induced bias is a random number. At first, Bagajewicz[26] proposed to calculate the induced bias as the maximum possible value, but more recently, Bagajewicz[30] and Bagajewicz and Nguyen[31] proposed to calculate the induced bias as the expected value of all possible values instead, which is more realistic, and used a Monte Carlo simulation-based procedure to obtain such expected value as well as some approximate method to perform the necessary integral calculations numerically.

Economic Value of Accuracy

Bagajewicz et al.[32] introduced the theory of economic value of precision and developed formulas for assessing downside financial loss incurred by production loss. They argued that, due to inaccuracy (caused by random errors) of the estimator of a product stream flow rate, there is a finite

probability that the estimator is above the target but in fact the real flow is below it. In such situation and under the assumption that the operators did not make any correction to the production throughput set point when the estimator suggested that the targeted production has been met or surpassed, the production output will be below the target and financial loss occurs. The financial loss under simplified assumptions of negligible process variations and normal distributions of the process variation and the measurements was found to be $\mathrm{DEFL} = 0.19947 \times K_s \times T \times \hat{\sigma}_p$ where K_s is the cost of the product (or the cost of inventory) and T is the time window of analysis (Bagajewicz et al.[32]).

Using the same concept of downside financial loss, Bagajewicz[27] extended the theory of economic value of precision to include the effect of (induced) bias, namely the economic value of accuracy. The expression for financial loss DEFL considering bias is given by (Bagajewicz[27]):

$$\mathrm{DEFL} = \Psi^0 \mathrm{DEFL}^0 + \sum_i \Psi^1_i \mathrm{DEFL}^1\big|_i + \sum_{i_1,i_2} \Psi^2_{i_1,i_2} \mathrm{DEFL}^2\Big|_{i_1,i_2} + \ldots \sum_{i_1,i_2,..,i_N} \Psi^n_{i_1,i_2,..,i_N} \mathrm{DEFL}^N\Big|_{i_1,i_2,..,i_N} \quad (3)$$

In this equation, $\Psi^n_{i_1,i_2,..,i_N}$ and $\mathrm{DEFL}^N\Big|_{i_1,i_2,..,i_N}$ are the average fraction of time the system is in the state containing n gross errors i_1, i_2,... ,i_N and its associated financial loss, respectively. The financial loss for system containing n biases i_1, i_2,... ,i_N are integrals that do not have analytical solution:

$$\mathrm{DEFL}^N\big|_{i_1,i_2,...,i_N} = \frac{K_s T}{2} \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \ldots \int_{-\infty}^{+\infty} f(\theta_1, \theta_2, \ldots \theta_n) d\theta_1 d\theta_2 \ldots d\theta_n$$

where θ_1, θ_2, ... θ_n are bias sizes. A detailed expression and procedure to calculate the financial losses $\mathrm{DEFL}^N\big|_{i_1,i_2,..,i_N}$ can be found in Nguyen et al.[28] In turn, if the probability of failure of sensor "i_1" is $f_{i_1}(t)$, then the associated fraction of time that the system is in that specific state (containing the specific set of gross errors i_1, i_2,... ,i_N) is $\Psi^n_{i_1,i_2,...,in_b} = f_{i_1}(t)..f_{i_n}(t) \prod_{s \neq i_1,...s \neq i_n} [1 - f_s(t)]$.

The instrumentation upgrade (adding new sensors) would reduce the individual financial loss (i.e., $\mathrm{DEFL}^0, \mathrm{DEFL}^1\big|_i, \mathrm{DEFL}^2\big|_{i_1,i_2}$...) in Eq. 3 (the main reason is that there are more measurements to improve precision of estimators and detect biases), the result is that financial loss would decrease.

Applications of the theory of economic value of precision/accuracy for the determination of economical benefit of instrumentation upgrade (IU) were shown by Bagajewicz et al.[32] and Bagajewicz.[27] The economical benefit of an instrumentation upgrade was calculated as the difference in downside financial loss (DEFL) before and after such upgrade. The net present value of instrumentation upgrade was then given by:

$$\mathrm{NPV} = d_n\{\mathrm{DEFL}(\text{before } IU) - \mathrm{DEFL}(\text{after } IU)\} - \text{cost of } IU \quad (4)$$

where d_n is sum of discount factor for n years. The cost can be the cost of purchasing of new sensor (when adding new sensors) or the cost of license (when installing data reconciliation software). A large value of the net present value of instrumentation upgrade may justify this type of investment. Case studies on the value of performing data reconciliation as well as savings of adding new sensors at selected locations to the sensor network of a crude distillation unit were provided by Bagajewicz et al.[32] and Bagajewicz.[27]

The financial loss $\mathrm{DEFL}^N\big|_{i_1,i_2,..,i_N}$ corresponding to the presence of a specific set of gross errors i_1, i_2,... ,i_N can be evaluated using two methods (Nguyen et al.[28]):

- Approximate method.
- Monte Carlo simulation.

The principle of the approximate method is to partition the space of variables into several subspaces. In some subspaces, the expression for financial loss $\mathrm{DEFL}^N\big|_{i_1,i_2,...,i_N}$ can be evaluated analytically, whereas in the others the expression has to be evaluated approximately (Nguyen et al.[28]). The partition of the space of variables is illustrated in Figure 1 for the case of two biases present in the system.

In the region where both biases are detected, the expression for financial loss can be calculated analytically, whereas in the others, an approximate scheme is used to evaluate the expression (Nguyen et al.[28]).

The Monte Carlo method (for calculating $\mathrm{DEFL}^N\big|_{i_1,i_2,...,i_N} = \frac{K_s T}{2} \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \ldots \int_{-\infty}^{+\infty} f(\theta_1, \theta_2, \ldots \theta_n) d\theta_1 d\theta_2 \ldots d\theta_n$) comprises three steps:

(i) Sampling the independent variables (the biases sizes θ_1, θ_2, ... θ_n) according to their distributions, which are assumed to be normal distributions;

(ii) Evaluating the integrand functions;

(iii) Evaluating the value of the integral (i.e., $\mathrm{DEFL}^N\big|_{i_1,i_2,...,i_N}$) as the mean value of the values of the integrand functions after a certain number of samplings (e.g., 1000 samplings). More details can be found in Nguyen et al.[28]

Similarly, the stochastic accuracy can be evaluated using the same two methods, the approximate method and a Monte Carlo simulation, with slightly different integrand functions in the case of the approximate method and evaluating departures from true values instead of economic losses in the Monte Carlo simulations.

Both the stochastic accuracy and financial loss are indicators of the quality of measurements: smaller values of stochastic accuracy and financial loss indicate better quality of data (i.e., a better sensor network that provides more accurate reconciled data).

Connections Between Software Accuracy and Expected Economic Loss

Because software accuracy is defined as precision plus induced bias, the requirement on accuracy value encompasses the requirements on precision, gross errors detectability, and gross errors resilience. More specifically, a sensor network that renders good (small) software accuracy for variables of interest needs to possess all of the followings:

(i) Good precision of estimators of key variables.

(ii) Good level of redundancy (i.e., enough measured variables) to detect biases so that undetected biases would have

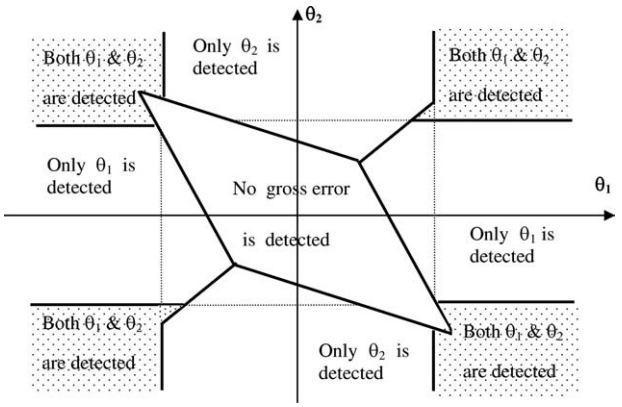


Figure 1. Different regions when two gross errors are present in the system.

small magnitudes; this property is directly related to gross errors detectability.

(iii) Smearing effect of undetected biases on estimators of key variables is limited (such that estimation accuracy is small even though undetected biases are large); this property is directly related to gross errors resilience.

These needed network's capabilities generally require a good level of software redundancy (i.e., more sensors than the number of key variables). To improve estimation accuracy, it is usually needed to use more sensors. The same thing is stated for financial loss, that is, sensor network would have small financial loss if it possesses the three aforementioned properties and it is necessary to use more sensors to reduce financial loss.

The exception to this generalization does exist. Indeed, there exists a situation in which the undetected biases can be very large, for example, two gross errors cancel out each other such that these two biases are undetected (by using measurement test) no matter how big they are. This phenomenon is explained by the theory of gross errors equivalency (Bagajewicz and Jiang[33]). To illustrate it, consider the system shown in Figure 2. The two biases in S_2 and S_3 cannot be detected (no matter how big they are) if they are equal but in opposite sign (because the material balance is satisfied in such case). The region of undetected biases for such case is shown in Figure 3.

However, gross errors are not unbounded. If a bias in a measurement passes a certain threshold, which is usually a certain percentage of the normal value of the variable, by common sense the operators can tell that there is bias in the measurement. The threshold above which bias is detected by the operators' judgment is used as (upper) limit for bias. Figure 4 shows such limits for both cases.

Thus, the accuracy calculations need to be made using additional box constraints on each variable.

$$-\delta_{\max,s} \le \delta^{(p)}_{\text{crit},s} \le \delta_{\max,s} \quad \forall s \tag{5}$$

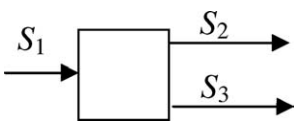


Figure 2. Illustrated example.

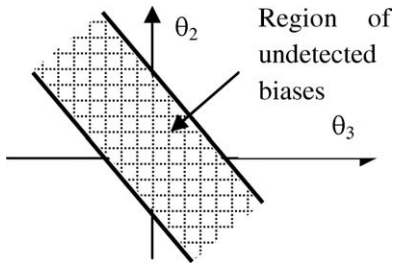


Figure 3. Illustration of biases equivalency.

Consider now a procedure to design a sensor network based on adding one instrument at a time, like the tree search procedure. If a new measurement forms such a set of (unbounded) undetected gross errors with existing measurements (whereas the original network does not have that set), the software accuracy and financial loss increase when this new measurement is added to the network. We call this the "Atypical" case because occasionally one might see deterioration in accuracy when sensors are added. The "Atypical case" and the "Typical case" of software accuracy and financial loss as function of the number of measurements are shown in Figure 5.

As can be seen from Figure 5:

- The jumps (steep slopes) in Figure 5 corresponding to the case where the newly added measurement contributes significantly to the process monitoring capabilities of the sensor network (e.g., observability and redundancy of key variables). On the other hand, if "meaningless" measurement (that contributes almost nothing to the process monitoring capabilities of the sensor network) is added, the accuracy and financial loss are almost unchanged.
- Generally, adding sensors improves accuracy and financial loss.
- If bias in the newly added sensor is very difficult to be detected, accuracy and financial loss would increase when adding that sensor. However, continue adding more sensors would again improve accuracy and financial loss.

Software accuracy can be used as a constraint in the commonly used cost-optimal SNDP.

Value-Optimal SNDP

The problem formulation for cost based accuracy-constrained SNDP is as follows:

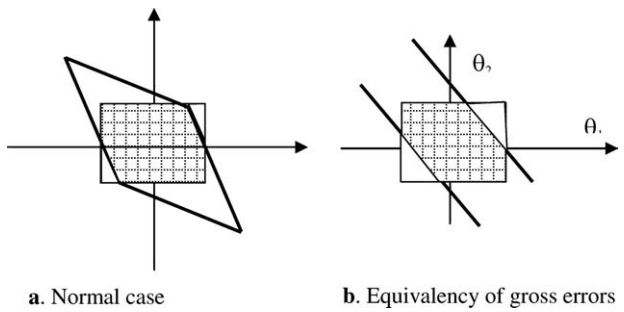
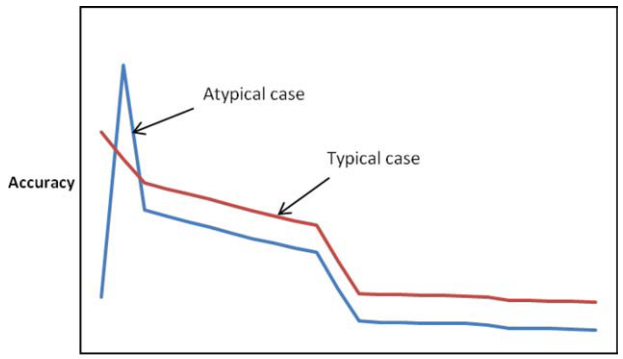
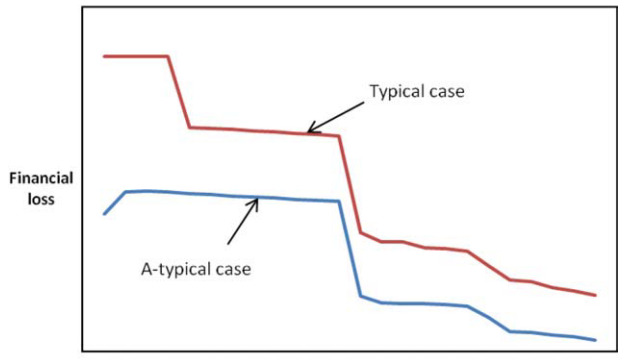


Figure 4. Box and rhombus constraints defining undetected biases.



a. Accuracy vs. number of sensors



b. Financial loss vs. number of sensors

Figure 5. Accuracy and financial loss as function of number of sensors.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

$$\begin{aligned} &\text{Min} \sum_{\forall i} c_i q_i \\ &\text{s.t.} \\ &a_i(\boldsymbol{q}) \le a_i^* \qquad \forall i \in N_S \\ &q_i = 0, 1 \qquad \forall i \end{aligned} \tag{6}$$

where q_i is the element of the vector of binary variable indicating that a sensor is located in variable i, c_i is the cost of such a sensor and N_s represents the set of variables where an accuracy specification $a_i(\boldsymbol{q})$ is required, and a_i^* the associated threshold values. To this formulation, other constraints can be added (precision, reliability, etc.) specifically, even though the accuracy usually contains all of them.

Problem (6) can be readily solved by using any suitable branch and bound method developed in our group (e.g., the cutset-based method or the breadth-first tree search method).

Departing from the above minimum cost paradigm, the proposed value-optimal problem formulation is as follows:

$$\begin{aligned} &\text{Max}\{V(\boldsymbol{q}) - c(\boldsymbol{q})\} \\ &\text{s.t.} \\ &c(\boldsymbol{q}) \le \boldsymbol{b} \end{aligned} \tag{7}$$

where $V(\boldsymbol{q})$ is the economic value of the data provided by the sensor network (function of measurement locations $\boldsymbol{q}$), $c(\boldsymbol{q})$ is cost of sensors, and b is limit on budget. If the budget limitation is not used, the problem becomes an unconstrained optimization problem where $\{V(\boldsymbol{q}) - c(\boldsymbol{q})\}$ is maximized. In turn, the value of a sensor network is given by

$$\begin{aligned} V(\boldsymbol{q}) &= \{\text{DEFL(no sensor)} - \text{DEFL(with sensors)}\} \\ &= \text{RDEFL} - \text{DEFL}(\boldsymbol{q}) \end{aligned} \tag{8}$$

The financial loss when there is no sensor is a large value, denoted as RDEFL (a reference value). Thus, $\{V(\mathbf{q}) - c(\mathbf{q})\}$ is given by:

$$\{V(\boldsymbol{q}) - c(\boldsymbol{q})\} = \{\text{RDEFL} - \{\text{DEFL}(\boldsymbol{q}) + c(\boldsymbol{q})\}\} \tag{9}$$

Thus, maximizing value minus cost is equivalent to minimizing financial loss plus the cost of the sensor network $(\text{DEFL}(\boldsymbol{q}) + c(\boldsymbol{q}))$.

As discussed above, **DEFL**$(\boldsymbol{q})$ is not monotonic with the addition of sensors, but cost is. Thus, the sum of those is not monotonic with the addition of sensors and many local minima could be observed in a method that relies on the addition of sensors as a procedure (tree search methods). We note that it is not simple to write Karush–Kuhn–Tucker (KKT) conditions and solve this problem using conventional mathematical programming techniques. In fact, a numerical method in the form of approximate method or Monte Carlo method, as discussed above, must be used to calculate the objective function (more specifically, to calculate the financial loss). Thus, the only applicable methods are the tree enumeration method and GA, a very popular stochastic search method to solve combinatorial optimization problem.

It can be seen that in the above optimization problem, if the sensors are expensive and/or K_s value (cost of product) is small, then the cost term dominates the financial loss term and the optimal solution would contain a small number of sensors. On the other hand, if the products are expensive (i.e., large K_s values) and/or the sensors are cheap, the financial loss factor dominates the cost factor and the optimal solution would contain a large number of sensors. This means that at fixed sensors cost, if K_s value increases, the number of sensors in optimal network would increase as evidenced in the illustrated example shown next.

Before we proceed to present computational methods to solve the proposed problems, we discuss the connection between this new approach (value-optimal SNDP without any constraint on performance target) and the traditional cost-optimal approach that minimizes sensors cost subjected to requirements on performance target.

Connection Between the Value and the Cost-Optimal Paradigms

The connection is discussed through an illustrated example shown below:

Example 1

Consider the following process example, which is shown in Figure 6. The flow rate, precision, and cost of sensors for example 1 are shown in Table 1. The proposed SNDPs were

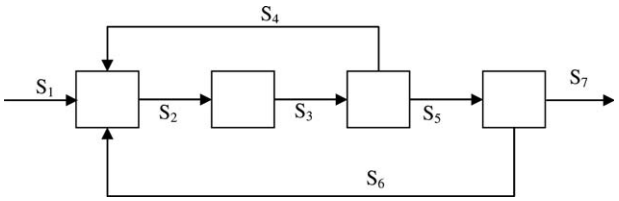


Figure 6. Example process.

Table 1. Data for Example 1

Stream	Flow Rates	Sensor Precision (%)	Sensor Cost
S_1	100	2	55
S_2	140	2	40
S_3	140	2	60
S_4	20	2	50
S_5	120	2	45
S_6	20	2	55
S_7	100	2	60

implemented in Fortran running on a 2.8-GHz Intel Pentium CPU, 1028-MB RAM PC. This example problem is solved using the tree search (branch and bound) method using individual measurements (Bagajewicz[34]) without stopping criterion, so the obtained solution is guaranteed to be optimal solution.

Traditional Cost-Optimal Approach not Based on Accuracy. The design specifications for four design cases are shown in row 2 to row 7 of Table 2. The last two rows show the optimal solution (optimal measurements location and optimal cost).

- For design case 1.1c: in addition to the specifications used in design case 1.1b, it is required that biases whose magnitude is greater than four times the standard deviation be detected. It should be noted that, although design case 1.1.c uses more specifications than design case 1.1b, the optimal solution is unchanged.
- For design case 1.1d: in addition to the specifications used in design case 1.1c, it is required that the induced biases in key variables caused by any biases in the system be less than 3 times the standard deviation.
- As one increases the level of specifications (e.g., more requirements to be satisfied), the obtained (feasible) solution would contain more sensors. This could be adding new sensor (e.g., $\{S_1, S_6, S_7\}$ ==> $\{S_1, S_5, S_6, S_7\}$) or using a different set of sensors ($\{S_1, S_5, S_6, S_7\}$ ==> $\{S_1, S_2, S_4, S_5, S_6\}$).
- It should be noted that in design case 1.1a where only observability is required, because a good level of precision is used (estimation precision is 1.5% vs. sensor precision = 2%), the number of measurements in optimal sensor network is more than the number of key variables (the optimal sensor network renders redundancy of level one for key variables).

Cost-Optimal SNDP with Requirement on Accuracy. The SNDP requesting a satisfactory accuracy value of key variables is illustrated next. The requirement on accuracy ($a_i(\mathbf{q}) \leq a_i^*$) is the only constraint in the problem. The results are shown in Table 3. The design specifications are shown in rows 2 and 3 of Table 3, the optimal solutions are shown in row 5 (optimal measurement placement) and row 6 (optimal cost). The (software) accuracy values of key variables corresponding to the optimal solutions are shown in row 4.

- For the design cases 1.2a–1.2e, the desired value of accuracy decreases (from 4.0 to 1.5), which requires more sensors to be used. In design case 1.2a, basically only observability is required for the two key variables S_1 and S_5. In design cases 1.2b and 1.2c, the obtained optimal solutions render redundancy of level one for key variables (the two key variables are still observable if one removes any one sensor out of the three-sensor solutions). It can be seen that the optimal solution in the design case 1.2c is the same as the solution obtained in design case 1.1a (column 2, Table 2) where a good level of precision (1.5%) is used.
- When a smaller accuracy threshold (design case 1.2d) is required, both redundancy (of key variables) and gross error detection capability of the network are required; hence more sensors need to be used. It can be seen that the optimal solution in the design case 1.2d is the same as the solution obtained in design case 1.1c (column 4, Table 2) where both estimation redundancy and gross error detection capability are required.
- Design case 1.2d is the extreme case where the required accuracy threshold is so small such that all sensors need to be used to meet the requirement.

Value-Optimal SNDP. The SNDP simultaneously minimizing financial loss and cost of a sensor network is illustrated next. This problem does not have any constraint. The economic parameters used in the expressions to evaluate financial loss are as follows: the time window of analysis T is 30 days (this is based on the argument that, by means of production accounting calculation every month, one can detect the loss in production that has been covered by biased measurement); the cost of product K_s (or cost of inventory) for the two key variables S_1 and S_5 are shown in row 3 of Table 4. The financial losses of the optimal sensor networks are shown in row 6 of the Table 4.

Table 2. Results for Example 1—Cost-Optimal SNDP

Case Study	1.1a	1.1b	1.1c	1.1d
Key variables	S_1 and S_5	S_1 and S_5	S_1 and S_5	S_1 and S_5
Requirement	Observability	Redundancy	Redundancy and error detectability	Redundancy and error detectability and resilience
Precision thresholds	1.5%	1.5%	1.5%	1.5%
Residual precision thresholds		4%	4%	4%
Error detectability thresholds			4	4
Error resilience threshold				3
Measured variables	S_1, S_6, S_7	S_1, S_5, S_6, S_7	S_1, S_5, S_6, S_7	S_1, S_2, S_4, S_5, S_6
Sensors cost	170	215	215	245

Table 3. Results for Example 1—Cost-Optimal SNDP with Accuracy Constraint

Case Study	1.2a	1.2b	1.2c	1.2d	1.2e
Key variables	S_1 and S_5	S_1 and S_5	S_1 and S_5	S_1 and S_5	S_1 and S_5
Accuracy thresholds (%)	4	3	2	1.8	1.5
Accuracy value (%)	$a_{s_1} = 3.36$ $a_{s_5} = 2.85$	$a_{s_1} = 2.22$ $a_{s_5} = 1.99$	$a_{s_1} = 1.90$ $a_{s_5} = 1.81$	$a_{s_1} = 1.65$ $a_{s_5} = 1.49$	$a_{s_1} = 1.499$ $a_{s_5} = 1.27$
Measured variables	S_1, S_6	S_1, S_5, S_6	S_1, S_6, S_7	S_1, S_5, S_6, S_7	All variables
Sensors cost	110	155	170	215	365

As can be seen from Table 4:

- When the cost of product K_s increases, financial loss increases.
- As explained above, when K_s value increases, the number of sensors in optimal network increases.
- In design case 1.3a, K_s value is small: the cost factor dominates the financial loss factor => cost needs to be minimized and the optimal network contains only enough sensors to guarantee observability of key variables. This optimal network is the same as the optimal network obtained in design case 1.2a where only observability of key variables is required (by using a large accuracy threshold).
- In the opposite case, design case 1.3d, K_s value is large: financial loss needs to be minimized and the optimal network contains all sensors (the same result as the result obtained in design case 1.2e). This is an extreme case.
- In design cases 1.3b and 1.3c, K_s value is moderate, the optimal networks contain enough sensors that can guarantee some degree of estimation redundancy and gross error detection capability. The optimal networks in these two design cases are respectively the same as the networks obtained in the two design cases 1.2c and 1.2d and also the two design cases 1.1a and 1.1c as shown in columns 4 and 5 of Table 3 (SNDP with accuracy constraint) and columns 2 and 4 of Table 2 (traditional cost-optimal SNDP). These two optimal networks have good process monitoring capability (good accuracy value and good gross error detection capability).

The above results clearly indicate that the entities software accuracy and the associated economic value (financial loss DEFL) encompass all the mentioned process monitoring performance measures of sensor network (precision, error detectability, etc.): if one reduces accuracy threshold or if the products are more pricey (larger K_s value), one asks for higher performance of sensor network (e.g., the sensor network needs to satisfy error detection capability in addition to observability and redundancy of key variables). The nice thing about the value-optimal approach is that the users need not to worry about what threshold values to be used in the constraints in traditional cost-optimal SNDP: a small K_s value would ask for decent performance of sensor network while a large K_s value asks for sensor network with very high performance, including the extreme case where all sensors are measured.

The rest of this article focuses on the computational methods to solve the value-optimal SNDP (as mentioned above, the accuracy-constrained SNDP is a constrained optimization problem that can be readily solved by using any appropriate branch and bound method that has been developed, for example, the cutset-based method proposed by Gala and Bagajewicz[11]).

Two methods that do not guarantee global optimality are proposed in this work for problems in which exhaustive enumeration is impractical. These methods are: GA and Cutset-based method.

Genetic algorithm

The proposed optimization problem is amenable to the use of a standard GA because:

- The problem is a combinatorial optimization problem involving binary variables (vector $\boldsymbol{q}$).
- There are no constraints, although when capital budget constraints are added a penalty function can be added to the evaluation of fitness.
- The objective function is a complicated function with many extrema.

In brief, the GA method is based on the principles of genetics, natural selection and evolution; it "allows a population composed of many individuals to evolve under specified selection rules to a state that maximizes the "fitness," i.e., minimizes the cost function" (Haupt and Haupt[35]). The algorithmic procedure and detailed description of the well-known GA method can be found in various textbooks such as the Haupts' book.[35]

The GA is briefly described as a seven-step procedure as follows:

(1) Variable encoding and decoding: this step involves the conversion (i.e., encoding) of the values of decision variables into an appropriate representation (a chromosome). Decoding is the reverse process of encoding.

(2) Initialization of population: this step involves randomly generating a population of N chromosomes. The size of population, N, is a GA parameter.

(3) Natural selection: this step involves three operations: (i) evaluating the cost function corresponding to each chromosome/individual in the population, (ii) sorting the population in descending order of "fitness," (iii) selecting a portion of population with good fitness value to keep and discarding the rest.

(4) Selection: selecting and pairing the retained (survived) chromosomes to produce offspring for the next generation.

Table 4. Results for Example 1, Value-Based SNDP

Case Study	1.3a	1.3b	1.3c	1.3d
Key variables	S_1 and S_5	S_1 and S_5	S_1 and S_5	S_1 and S_5
K_s value	$K_{s_1} = 2$ $K_{s_5} = 2$	$K_{s_1} = 10$ $K_{s_5} = 10$	$K_{s_1} = 30$ $K_{s_5} = 20$	$K_{s_1} = 60$ $K_{s_5} = 50$
Measured variables	S_1, S_6	S_1, S_6, S_7	S_1, S_5, S_6, S_7	All
Sensors cost	110	170	215	365
Financial loss	78.6	219.9	451.8	824.9

(5) Mating: offspring of the paired chromosomes (parent) are produced through the crossover process whereby the parent's genetic codes are passed on to the offspring.

(6) Mutation: random mutations alter a certain percentage of the bits in the list of chromosomes. Mutation points are randomly selected from the population; with each mutation point, changing a 1 to a 0 and visa versa. The number of mutation points is defined by mutation rate, which is the fraction of the number of mutation points divided by the total number of bits in the population.

(7) Convergence: after the mutation step, a next generation population is generated, which contains new chromosomes. The same cycle (steps 3–6) is repeated unless convergence criterion is met, which is to terminate the GA procedure if the best objective value obtained in each iteration does not change after a predetermined number of iterations.

The parameters involved in the GA method are the size of population, the portion of population to keep, the mutation rate and the selection and crossover methods. The methods for the GA operators and the values are intuitively chosen in accordance with the scale of the problem using the guidelines provided in the literature (Haupt and Haupt[35]). They are as follows:

- Selection: roulette wheel selection method.
- Crossover: two-point crossover method.
- Population size = 20.
- Fraction of population to keep = 0.5.
- Mutation rate = 0.2.

These parameters are used in the medium size problem shown in the illustrated example section. For bigger scale problems, a larger population size and a greater mutation rate should be used.

Cutset-based tree search method

The calculation procedure is based on a tree search procedure with branching and stopping criteria (Bagajewicz[34]) and is described next:

(1) Find all the cutsets of the process graph.

(2) Consider only cutsets that contain at least one key variable, put them to a list of cutsets.

(3) Remove key variables out of the cutsets in the list and consider them as separate cutsets, e.g., if [1 2 3 4] is a cutset and "1" and "3" are key variables then consider [1], [3], and [2,4] as separate cutsets.

(4) Sort these cutsets in ascending order of their cost (cost of a cutset is equal to sum of the costs of the sensors placed on the streams of that cutset).

(5) Start with the root node with no cutsets being added, i.e., $\boldsymbol{t} = \{0, 0, 0\ldots\}$, trivially infeasible.

(6) Use branching criterion to develop branches of the tree (add cutsets to vector $\boldsymbol{t}$).

(7) While performing the branching criteria, if any set of measurements has already been evaluated in previous nodes, that node is not continued. This occurs frequently because one set of measurements can be a result of the union of different sets of cutsets.

(8) Continue adding cutsets until the stopping criterion is met. In such case, the algorithm backs up two levels and develops the next branch.

Branching criterion

A cutset is added in the direction of minimum cost, that is, the newly added cutset is chosen such that the cost obtained by its union with the existing active cutsets is minimum.

An alternative branching has also been investigated, which is choosing cutsets in the direction of minimizing the objective function. It is found that this branching criterion requires much longer computational time than the other (direction of minimum sensors cost). In fact, for the small scale example given above (Figure 6), this branching criterion requires roughly 10 times more computational time than the other criterion. For medium or large-scale problems, the difference is much larger. This is because the calculation of financial loss is an intensive computation duty, especially for middle or large-scale problems.

The task remaining is to find a proper stopping criterion.

Stopping criteria

In the branch-and-bound method, in each node of the search tree, it is necessary to find the lower bound for the best solution obtainable if continuing exploring down the branch of the tree. If that bound is not better than the current best solution (the incumbent) obtained so far, stop exploring down the branch. Unless the bound is obvious, it is found by solving relaxation subproblems (e.g., linear programming (LP)-relaxation, Lagrangean relaxation) in the subspace of variables. Unfortunately, none of the established techniques to find the bound is applicable to our problem; the main reason is that there is no explicit expression for the objective function.

The proposed stopping criterion is as follows: In each node, the two terms ΔD and ΔC are defined and calculated as follows:

- Change in financial loss: ΔD = DEFL (current node) − DEFL (sensor network with maximum number of sensors).
- Change in cost: ΔC = Cost (sensor network with maximum number of sensors) − Cost (current node).

The change in financial loss (ΔD) indicates the maximum gain in financial loss, whereas the change in cost (ΔC) indicates the maximum cost incurred if one continues exploring down the tree from the current node. It can also be shown that if $\{\Delta C - \Delta D\}$ of current node $> \{\Delta C - \Delta D\}$ of previous node then the objective value of current node $<$ the objective value of the previous node. The reason why ΔD and ΔC are used is illustrated in Figures 7 and 8, where "MNS" is used to denote the network with maximum number of sensors (i.e., all sensors are used).

One should always start exploring the branch with nodes that have $\Delta D > \Delta C$ or $\{\Delta C - \Delta D\} < 0$ (in region I); the relationship $\Delta D > \Delta C$ implies that one can reduce the objective function if continuing exploring down the tree.

Thus, ΔD and ΔC are used because:

(i) The optimal solution cannot be in the region where not all key variables are observable (region I, Figure 7), which always has $\Delta D > \Delta C$.

(ii) The relationship $\Delta D > \Delta C$ implies that there is high potential of reducing the objective function when exploring down the tree; if $\Delta D < \Delta C$: less potential.

(iii) If $(\Delta C - \Delta D)$ of node 1 $>$ $(\Delta C - \Delta D)$ of node 2 then objective value of node 1 $<$ objective value of node 2.

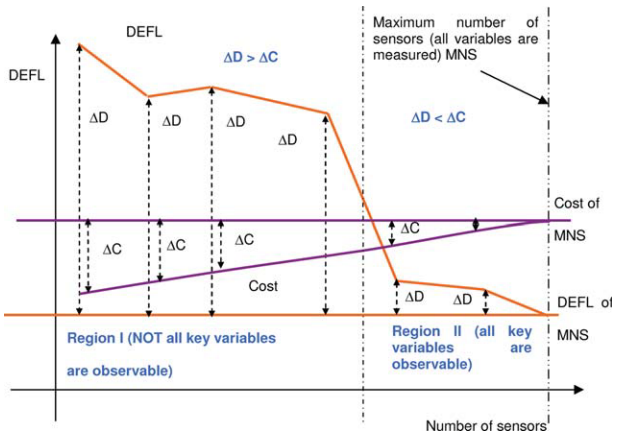


Figure 7. Differentiation of regions using ΔD and ΔC.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Using this relationship, with reference to Figure 8, we would have objective values of nodes $B,C,D <$ objective values of all the nodes that have $(\Delta C - \Delta D) < 0$ (the region on the left hand side).

The proposed stopping criterion is:

- Exploring down the branch until $\Delta D < \Delta C$.
- When $\Delta D < \Delta C$, explore further down the branch until objective value of current node > objective value of previous node.

The essence of this proposed stopping criterion is, in a branch of the tree, locating a local minimum in the region of less potential of reducing objective function.

We now investigate the possibility that the global optimal solution is missed because the proposed stopping criterion stops the tree search before it reaches global optimal. This is illustrated by in Figure 9 (pathway A)

We have a conjecture (based only on testing) that there always exists a monotonic pathway to reach global optimum (pathway B in Figure 9). A proof of this assertion is left for future work. The supporting reasons for the conjecture are:

- A union of variables (streams) is a result of many combinations (unions) of cutsets. This fact implies that, when

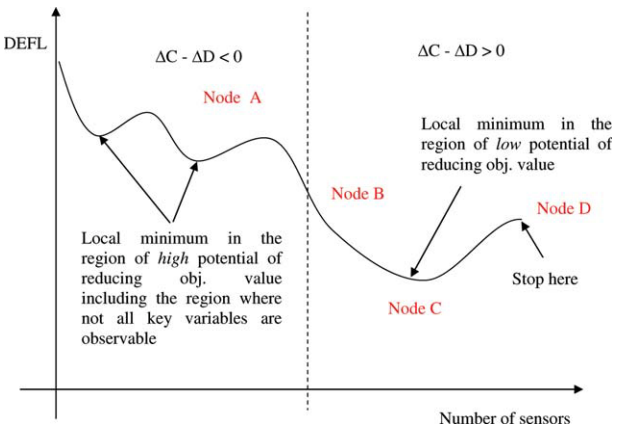


Figure 8. Use of ΔD and ΔC in stopping criterion.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

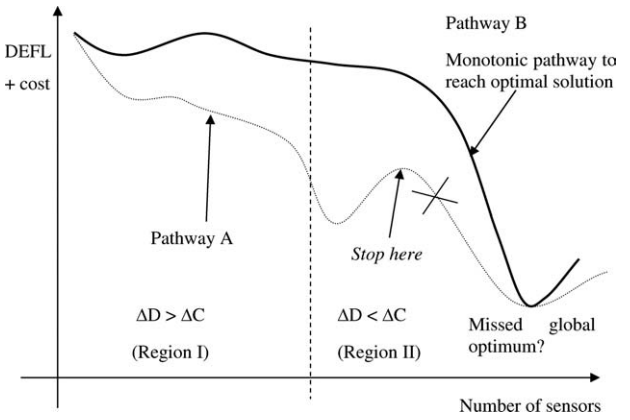


Figure 9. Illustration of missing optimal solution because of stopping criterion.

cutsets are used in the tree search procedure, a specific set of active streams (measurements) can be reached by following many pathways (branches) in the tree. Table 5 shows an estimate of how many pathways (using cutsets) to reach a specific set of active streams (i.e., measurements location) for the Madron process problem (shown in next section).

- In most of the cases one can find a pathway in which the objective function is a monotonic decreasing function until it reaches the optimal solution (or at least objective function is monotonic decreasing in the region $\Delta D < \Delta C$ where the stopping criterion is considered). Note that changing the pathway is actually following another branch in the tree. We do not claim that one can always find such a pathway because there is no mathematical proof for this, but we have not found a counter example in which the global solution cannot be reached by following any branch or pathway using the stated stopping criteria.

The third row of Table 5 shows the number ($N1$) of possible combinations (sets) of cutsets from a given number of cutsets, whereas the fourth row shows the number ($N2$) of candidate solutions (i.e., measurements locations) resulting from the same given list of cutsets. The ratio $N1/N2$ is an indicator of how frequently the situation that two sets of cutsets result in the same measurements location (by union operation) can occur. For example, if the ratio is 100, then among 100 possible combinations of cutsets, only one combination leads to a candidate solution, the remaining 99 combinations are disregarded because they result in the same measurements location. This also means that expectedly there are 100 pathways to reach a specific set of active measurements. The information shown in Table 5 is obtained from the Madron example (containing 24 streams, shown in next section).

Table 5 shows that only roughly 50,000 candidate solutions (each solution is a specific set of measurement locations, for comparison, the total number of such set of measurement locations is $2^{24} - 1$) resulted from the $(2^{100} - 1)$ possible combinations of cutsets. This result reveals that the number of pathways (built on cutsets) to reach a specific set of measurements is very large.

All these discussions make us conclude that:

- Because there are so many pathways to reach a candidate solution, if the global optimal solution is not reachable

Table 5. Estimate of Pathways (Built on Cutsets) to Reach a Specific Set of Measurements

Case	1	2	3	4
Key variables	$\{S_1, S_9, S_{14}\}$	$\{S_1, S_5, S_{22}\}$	$\{S_1, S_5, S_{24}\}$	$\{S_1, S_7, S_{24}\}$
Number of key variables	3	3	3	3
Number of cutsets containing at least one key variable	99	97	102	108
Number of possible combinations of cutsets ($N1$)	$2^{99} - 1$	$2^{97} - 1$	$2^{102} - 1$	$2^{108} - 1$
Number of candidate solutions ($N2$)	46,042	64,781	39,552	38,365
$N1/N2$	1.38×10^{25}	2.45×10^{24}	1.28×10^{26}	8.46×10^{27}

in a pathway (because that pathway is not monotonic), it would be reachable in another pathway. Thus, the chance of finding global optimal solution is very high.

- The bad side of this fact is that the stopping criterion may not have any effect at all, that is, one candidate solution if not reachable in a pathway can still be reachable in another pathway. The result is that the number of candidate solutions explored is equal (or almost equal) for both cases: with and without stopping criterion. The obtained results from the Madron example confirm this speculation.
- Thus, it can be predicted that the performance of the presented stopping criterion is not satisfactory: the reduction in computational time is insignificant (because the number of candidate solutions explored is almost equal for both cases: with and without stopping criterion) and there is no guarantee of optimality (although the chance of finding optimal solution is very high). However, it is the best criterion that we can find for this type of problem. Separate work[36] focuses on improving the efficiency of cutset-based method by using parallel computing rather than finding a better stopping criterion.

The performance of the proposed methods is shown in the following example.

Example 2

All of the proposed methods were implemented in Fortran. The exhaustive tree search, the GA method and the cutset-based method were run on a 2.8-GHz Intel Pentium CPU, 1028-MB RAM PC.

The flowsheet of the example, which was introduced by Madron and Veverka[3] is given in Figure 10. Madron and Veverka[3] did not report flow rates, so the flowrate values shown in Table 6 were taken from Bagajewicz.[34] The precision and cost of sensors are also given in Table 6.

Information used in the calculation of financial loss is as follows:

- Probability of sensors = 0.1 (for all sensors).
- Biases (in failed sensors) are assumed to follow normal distribution with zero means and standard deviations = 4.0 (for all sensors).
- Windows time of analysis $T = 30$ days.
- The K_s values (cost of product or cost of inventory) vary with design case studies, which are shown in Table 7.

The 10 design case studies together with the optimal solutions obtained by using the Cutset-based methods are shown in Table 7.

Exhaustive Tree search using individual sensors

When the tree search method is conducted using individual measurements (Bagajewicz[34]) instead of cutsets the procedure took a very long time. This is because the problem contains 24 streams, and hence, the total number of candidate solutions is $2^{24} - 1 = 16.78$ millions. In fact, after one month (30 days) of running time, the computational process was terminated. When stopped, the tree search explored only 4.19 millions of candidate solutions (and was able to identify the optimal solution shown in row 2 of Table 7), hence the estimated computation time of this method is 120 days (4 months). Computational time in other design case studies should be at the same magnitude with this computational time (120 days). Thus, this method is applicable for small scale problems only.

Exhaustive tree search using cutsets

The last four columns of Table 7 show details of the optimal solutions (number of sensors, their location, cost, and financial loss). These optimal solutions are obtained by using the Cutset-based tree search method described in the above section without a stopping criterion, which means that, in each design case all the candidates for optimal solution are explored, thus the identified solutions are global optima.

A few observations can be withdrawn from the above results:

- The locations of key variables can greatly affect the financial loss and the obtained optimal network as evidenced in design cases 2.1–2.6: all of these six design cases have three key variables with similar K_s values (only locations of key variables are different) but the number of sensors in optimal network can change significantly (from 4 to 12). It can be seen that if S_1 is a key variable (all design cases except cases 2.3, 2.4, and 2.5), the optimal network contains a large fraction of available sensors; if S_1 is not a key variable (design cases 2.3, 2.4, and 2.5) the optimal network contains a small number of sensors. The reason is that bias in measurement S_1 is more difficult to be detected than other

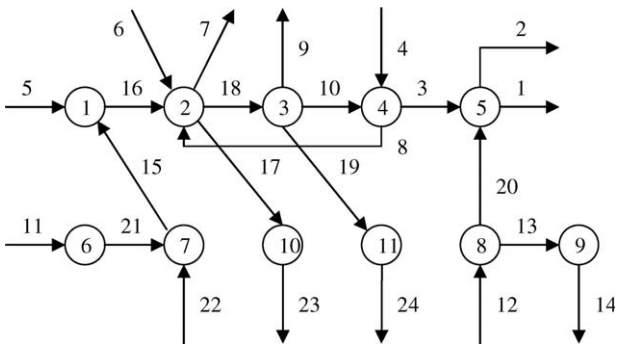


Figure 10. Flowsheet of Madron and Veverka's problem.

Table 6. Data for the Madron and Veverka's Problem

Stream	Flow	Sensor Cost	Sensor Precision (%)	Stream	Flow	Sensor Cost	Sensor Precision (%)
1	140	19	2.5	13	10	12	2.5
2	20	17	2.5	14	10	12	2.5
3	130	13	2.5	15	90	17	2.5
4	40	12	2.5	16	100	19	2.5
5	10	25	2.5	17	5	17	2.5
6	45	10	2.5	18	135	18	2.5
7	15	7	2.5	19	45	17	2.5
8	10	6	2.5	20	30	15	2.5
9	10	5	2.5	21	80	15	2.5
10	100	13	2.5	22	10	13	2.5
11	80	17	2.5	23	5	13	2.5
12	40	13	2.5	24	45	13	2.5

measurements, so more sensors are needed if S_1 is a key variable.

- As K_s values increase, the financial loss term dominates the cost term and optimal network would contain more sensors to reduce financial loss as evidenced in design cases 2.6 and 2.7 (same key variables, different K_s values) and design cases 2.8–2.10.
- There is a very high chance that all key variables appear in the optimal solutions (i.e., all key variables are measured): this is the case in all 10 design case studies under consideration.

Genetic algorithm

The performance of the GA method is shown in Table 8. In each design case, two attempts were made to solve the problem using GA, the better one among the two results obtained from these two attempts (in term of quality of objective value) is reported. In Table 8, the second and third columns show details (the number and the location of sensors) of the solutions obtained by GA method. The fourth column shows objective values of these solutions. For comparison, the optimal objective value (summation of sensors cost and financial loss shown in Table 7) is also shown in column five. The last column shows computational time of the GA method.

As shown in Table 8:

- Although the GA method does not guarantee optimality, it is able to locate optimal solution in two design cases 2.3 and 2.8. Moreover, in the other three design cases (2.7, 2.9, and 2.10), the best solutions obtained by GA are "very good": they are very near to the optimal solutions.
- Computational time of the GA method is acceptable: it solves this problem within an hour.

Table 7. Results for the Madron and Veverka's Problem

Case Study	Key Variables	K_s Value	Number of Sensors	Measured Variables	Sensors Cost	Financial Loss
2.1	1, 9, 14	$K_{s_1} = 25$ $K_{s_9} = 20$ $K_{s_{14}} = 20$	11	1, 2, 3, 4, 8, 9, 10, 12, 13, 14, 20	137	415.1
2.2	1, 5, 22	$K_{s_1} = 25$ $K_{s_5} = 20$ $K_{s_{22}} = 20$	11	1, 2, 3, 4, 5, 8, 10, 12, 13, 20, 22	158	471.4
2.3	2, 6, 24	$K_{s_2} = 25$ $K_{s_6} = 20$ $K_{s_{24}} = 20$	4	2, 6, 19, 24	57	400.1
2.4	4, 9, 23	$K_{s_4} = 25$ $K_{s_9} = 20$ $K_{s_{23}} = 20$	4	4, 9, 17, 23	47	283
2.5	4, 5, 24	$K_{s_4} = 25$ $K_{s_5} = 25$ $K_{s_{24}} = 45$	4	4, 5, 19, 24	67	527.1
2.6	1, 5, 24	$K_{s_1} = 25$ $K_{s_5} = 20$ $K_{s_{24}} = 20$	12	1, 2, 3, 4, 5, 8, 10, 12, 14, 19, 20, 24	175	498.7
2.7	1, 5, 24	$K_{s_1} = 45$ $K_{s_5} = 36$ $K_{s_{24}} = 45$	15	1, 2, 3, 4, 5, 8, 9, 10, 12, 13, 14, 18, 19, 20, 24	210	891.2
2.8	1, 7, 24	$K_{s_1} = 25$ $K_{s_7} = 20$ $K_{s_{24}} = 25$	12	1, 2, 3, 4, 7, 8, 10, 12, 13, 19, 20, 24	157	538.8
2.9	1, 7, 24	$K_{s_1} = 45$ $K_{s_7} = 40$ $K_{s_{24}} = 45$	19	1, 2, 3, 4, 6, 7, 8, 9, 10, 12, 13, 14, 16, 17, 18, 19, 20, 23, 24	251	859.3
2.10	1, 7, 24	$K_{s_1} = 80$ $K_{s_7} = 70$ $K_{s_{24}} = 80$	22	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 13, 14, 15, 17, 18, 19, 20, 21, 22, 23, 24	302	1471.8

Table 8. Performance of the GA Method

Case Study	Number of Sensors	Measured Variables	Objective Value	Optimal Objective Value	Computation Time
2.1	13	1, 2, 3, 4, 8, 9, 10, 12, 13, 14, 17, 20, 23	568.1	552.1	54 min
2.2	13	1, 2, 3, 4, 5, 6, 8, 10, 12, 13, 19, 20, 22	653.6	629.4	54 min
2.3	4	2, 6, 19, 24	457.1	457.1	25 min
2.4	6	4, 8, 9, 17, 21, 23	349.9	330	17 min
2.5	8	4, 5, 7, 13, 14, 19, 21, 24	614.8	594.1	20 min
2.6	15	1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 13, 18, 19, 20, 24	686.9	673.7	32 min
2.7	19	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 13, 14, 15, 18, 19, 20, 23, 24	1104.7	1101.2	55 min
2.8	12	1, 2, 3, 4, 7, 8, 10, 12, 13, 19, 20, 24	695.8	695.8	38 min
2.9	18	1, 2, 3, 4, 6, 7, 8, 9, 10, 12, 13, 14, 16, 17, 19, 20, 23, 24	1111.7	1110.3	39 min
2.10	23	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24	1775.5	1773.8	59 min

• In general, performance of the GA method is acceptable. Additionally, the GA method does not exhibit scaling problem (computational time does not increase exponentially with the size of the problem). To increase the chance of locating optimal solution, one can adjust the GA parameters (increase the size of population and/or mutation rate); or simply rerun GA many times (each GA run generally gives a different result).

Comparison with the cutset-based tree search method with stopping criteria

The performance of the cutset-based tree search methods are compared in Table 9.

Column 2 of Table 9 shows the number of cutsets (containing at least one key variable) in the corresponding design problems, the last column shows computational time of cutset-based method when stopping criterion is used. When stopping criterion is not used, the computational time is almost the same (the difference is usually not more than 5 min).

As can be seen from Table 9:

• When the stopping criterion is used, the cutset-based method is able to locate optimal solutions (although optimality is not guaranteed if the stopping criterion is used).

• The stopping criterion has ''little'' effect: the number of nodes explored and computational time when stopping criterion is used are almost unchanged when compared with the case stopping criterion is not used. Only in the design case 2.2 that there is a small difference in number of nodes explored between the two cases (results of other case studies (not shown here) of Madron example also testify this fact). Thus, it may be not necessary to use stopping criterion in cutset-based tree search method.

• Small size and medium size testing problems (the 7-stream example 1 and the 24-stream Madron problem) reveal that there is little different between the two cases: using stopping criterion and not using criterion. In other words, the stopping criterion does not help reduce computational time. It takes very long time to solve the large-scale problems (the ones with at least 40 streams) in both cases (with and without stopping criterion) so it is not known whether the stopping criterion indeed has some effect in reducing computational time for large-scale problems. For large-scale problems, we look for a better method rather than finding another stopping criterion. This is left for future work.

• It may also be not necessary to use branching criterion (just put cutsets in numbered order like [1] => [12] => [123]). The advantage of using branching criterion is that optimal solution is usually identified earlier than the case where branching criterion is not used: among the ten design case studies, there are six design cases where optimal solution is located within the first 20 nodes explored. This is very beneficial if the computational process has to be terminated halfway because the computational time becomes too long. The disadvantage is that using branching criterion costs more time.

• Performance of cutset-based method is acceptable for this medium size Madron problem. However, because this method exhibits scaling problem, it is not efficient enough for large-scale problems.

Conclusions

In this work, a value-paradigm to design sensor network is presented. This new approach simultaneously optimizes

Table 9. Performance of Cutset-Based Method

Case Study	Number of Cutsets	Number of Nodes Explored		Computational Time
		With Stopping Criterion	No Stopping Criterion	
2.1	99	46,042	46,042	9 h 4 min
2.2	97	64,773	64,781	11 h 44 min
2.3	108	38,070	38,070	4 h 20 min
2.4	105	28,178	28,178	2 h 45 min
2.5	105	34,134	34,134	3 h 31 min
2.6	102	39,552	39,552	7 h 57 min
2.7	102	39,552	39,552	7 h 56 min
2.8	108	38,365	38,365	8 h 2 min
2.9	108	38,365	38,365	8 h 3 min
2.10	108	38,365	38,365	8 h 1 min

performance and cost of sensor network. The connection between the value-paradigm and the traditional cost-optimal approach is also discussed and illustrated.

The value-optimal SNDP is a computationally challenging problem because this is an unconstrained optimization problem with an objective function that needs to be evaluated numerically.

The GA is the most commonly used optimization technique for this kind of problem; the GA is satisfactorily efficient enough but it does not guarantee optimality.

The cutset-based method without stopping guarantees optimal solution. If a stopping criterion is used, the cutset-based method is still able to locate optimal solution but optimal solution is not guaranteed and the stopping criterion has little effect in reducing computational time. Both methods solve small and medium size problems within an acceptable computational time. More efficient method that guarantees optimal solution for large-scale problems are needed and are left for future work. One such advance in computational efficiency is reported separately.[36]

Literature Cited

1. Bagajewicz M. *Process Plant Instrumentation: Design and Upgrade*. Boca Raton, FL: CRC press, 2000.
2. Ali Y, Narasimhan S. Sensor network design for maximizing reliability of linear processes. *AIChe J*. 1993;39:820–828.
3. Madron F, Veverka V. Optimal selection of measuring points in complex plants by linear models. *AIChe J*. 1992;38:227–236.
4. Meyer M, Le Lann J, Koehret B, Enjalbert M. Optimal selection of sensor location on a complex plant using a graph oriented approach. *Comput Chem Eng*. 1994;18:S535–S540.
5. Luong M, Maquin D, Huynh C, Ragot J. Observability, redundancy, reliability and integrated design of measurement systems. In: *Proceeding of 2nd IFAC Symposium on Intelligent Components and Instrument Control Applications,* Budapest, Hungary, June 8–10, 1994.
6. Chmielewski D, Palmer T, Manousiouthakis V. On the theory of optimal sensor placement. *AIChe J*. 2002;48:1001–1012.
7. Sen S, Narasimhan S, Deb K. Sensor network design of linear processes using genetic algorithms. *Comput Chem Eng*. 1998;22:385–390.
8. 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.
9. Carnero M, Hernandez J, Sanchez M, Bandoni A. On the solution of the instrumentation selection problem. *Ind Eng Chem Res*. 2005;44:358–367.
10. Kelly JD, Zyngier D. A new and improved MILP formulation to optimize observability, redundancy and precision for sensor network problems. *AIChe J*. 2008;54:1282–1291.
11. Gala M, Bagajewicz MJ. Rigorous methodology for the design and upgrade of sensor networks using cutsets. *Ind Eng Chem Res*. 2006;45:6687–6697.
12. Gala M, Bagajewicz MJ. Efficient procedure for the design and upgrade of sensor networks using cutsets and rigorous decomposition. *Ind Eng Chem Res*. 2006;45:6679–6686.
13. Nguyen DQ, Bagajewicz M. Design of nonlinear sensor networks for process plants. *Ind Eng Chem Res*. 2008;47:5529–5542.
14. Nguyen DQ, Bagajewicz M. New efficient breadth-first/level traversal tree search methods for the design and upgrade of sensor networks. *AIChe J*. In press.
15. Raghuraj R, Bhushan M, Rengaswamy R. Locating sensors in complex chemical plants based on fault diagnostic observability criteria. *AIChe J*. 1999;45:310–322.
16. 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.
17. 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.
18. 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.
19. Musulin E, Bagajewicz M, Nougues JM, Puigjaner L. Instrumentation design and upgrade for principal components analysis monitoring. *Ind Eng Chem Res*. 2004;43:2150–2159.
20. Bhushan M, Narasimhan S, Rengaswamy R. Robust sensor network design for fault diagnosis. *Comput Chem Eng*. 2008;32:1067–1084.
21. Kotecha PR, Bhushan M, Gudi RD. Constraint programming based robust sensor network design. *Ind Eng Chem Res*. 2007;46:5985–5999.
22. Kotecha PR, Bhushan M, Gudi RD. Design of robust, reliable sensor networks using constraint programming. *Comput Chem Eng*. 2008;32:2030–2049.
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. Bagajewicz M, Chmielewski D, Rengaswamy, R. Integrated process sensor network design. In: *Proceedings of the AIChe Annual Meeting*, Austin, Texas, 2004.
25. Narasimhan S, Rengaswamy R. Quantification of performance of sensor networks for fault diagnosis. *AIChe J*. 2007;53:902–917.
26. Bagajewicz M. On the definition of software accuracy in redundant measurement systems. *AIChE J*. 2005;51:1201–1206.
27. Bagajewicz M. Value of accuracy in linear systems. *AIChE J*. 2006;52:638–650.
28. Nguyen TDQ, Siemanond K, Bagajewicz MJ. Downside financial loss of sensor networks in the presence of gross errors. *AIChE J*. 2006;52:3825–3841.
29. Miller RW. *Flow measurement engineering handbook*. New York, USA: McGraw-Hill, 1996.
30. Bagajewicz M. On a new definition of a stochastic-based accuracy concept of data reconciliation-based estimators. In: *15th ESCAPE Proceeding (European Symposium on Computer-Aided Process Engineering*, Spain, 2005:1135–1141.
31. Bagajewicz M, Nguyen DQ. Stochastic-based accuracy of data reconciliation estimators for linear systems. *Comput Chem Eng*. 2008;32:1257–1269.
32. Bagajewicz M, Markowski M, Budek A. Economic value of precision in the monitoring of linear systems. *AIChE J*. 2005;51:1304–1309.
33. Bagajewicz M, Jiang Q. Gross error modeling and detection in plant linear dynamic reconciliation. *Comput Chem Eng*. 1998;22:1789–1810.
34. Bagajewicz M. Design and retrofit of sensors networks in process plants. *AIChE J*. 1997;43:2300–2306.
35. Haupt RL, Haupt SE. *Practical Genetic Algorithms*, 2nd ed. New Jersey, USA: Wiley-Interscience, 2004.
36. Nguyen TD, Bagajewicz M. Parallel computing approaches to sensor network design using the value paradigm. *Comput Chem Eng*. In press.

Manuscript received Mar. 9, 2010, and revision received Aug. 19, 2010.