PROCESS DESIGN AND CONTROL

Rigorous Methodology for the Design and Upgrade of Sensor Networks Using Cutsets

Mayur Gala and Miguel J. Bagajewicz*

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

Traditional sensor network design for data-reconciliation-based monitoring systems consists of optimally placing sensors to satisfy predefined precision goals as well as residual precision and gross error robustness constraints to achieve minimum sensor network cost. One existing formulation of the cost optimal sensor network design is mixed integer nonlinear [Bagajewicz, M. *AIChE J*. **1997**, 9, 2300.] which is solved using a tree search algorithm with some bounding properties. Although the tree search method guarantees global optimality, it fails to perform computationally well for medium size problems and fails altogether in large ones. Similar problems are found when using a MILP formulation [Bagajewicz, M.; Cabrera, E. *AIChE J*. **2001**, *48*, 2271.]. Other MINLP convex formulations [Chmielewski, D.; et al. Cost Optimal Retrofit of Sensor Networks with Loss Estimation Accuracy. Presented at the AIChE Annual Meeting, Dallas, TX, November, 1999. Chmielewski, D.; et al. *AIChE J*. **2002**, *48*, 1001.] have not been tested in large problems. Thus, the field has resorted to the use of genetic algorithms [Sen, S.; et al. *Comput. Chem. Eng.* **1998**, *22*, 385. Carnero, M.; et al. *Ind. Eng. Chem. Res.* **2001**, *40*, 5578. Carnero, M.; et al. *Ind. Eng. Chem. Res.* **2005**, *44*, 358.]. In this paper, we present an alternative sensor network design algorithm based on graph theory that guarantees global optimality and is faster than the existing tree searching approaches. The efficiency of the proposed algorithm, which is good for medium size problems, is illustrated.

Introduction

Of all the state variables and parameters that can be estimated in a plant, there is a subset for which sufficiently accurate estimates are needed. This information is used in process monitoring, control, quality assurance, production accounting, and fault detection and relies on the manipulation of information provided by sensors. Thus, the reverse engineering problem is one of determining which variable should be measured and with what precision so that a certain prespecified quality of data to fulfill the aforementioned goals is obtained.

In 1976, Vaclavek and Loucka[8] were the first to explore this problem using graph theory to ensure variable observability of a multicomponent flow network. In 1987, Kretsovalis and Mah[9] developed a combinatorial search algorithm for sensor network design based on the effect of the variance of measurements on the precision of reconciled values. In 1992, Madron and Veverka[10] proposed multiple Gauss Jordan elimination of the linear mass balance equation, minimizing the overall cost of the sensor and satisfying the condition that all key variables are observable. Ali and Narasimhan[11] introduced the concept of reliability of estimation of the variable and developed an algorithm that maximizes the reliability of the network. Later, Ali and Narasimhan[12] extended their previous work to redundant networks. Bagajewicz[1] introduced the robustness concepts of residual precision, error detectability, and resilience and formulated a mixed integer nonlinear programming (MINLP) problem to obtain cost optimal network structures for linear systems subject to constraints on precision and the aforementioned additional constrains on new robustness measures.

To solve the sensor location problem, different alternative objective functions were used by different researchers: minimum cost, maximum precision, and maximum reliability. Bagajewicz and Sánchez[13] showed that there exists a connection between the minimum cost and maximum precision model, and Bagajewicz and Sánchez[14] extended the idea showing the duality between the objectives of cost and reliability. Bagajewicz[15] reviewed all other extensions.

The method of tree enumeration by Bagajewicz[1] and the MILP procedures that followed[2−4] are the only procedures that guarantee optimality. They are, however, numerically inefficient. Indeed, they take too much computational time for medium size industrial type of problems. On the other hand, methods based on genetic algorithms, which perform reasonably well from the computational point of view but do not guarantee optimality, have been proposed.[5−7,16−18]

In the present article, we present a new tree enumeration approach based on branching and bounding that uses cutsets (a graph theory construct) to represent a solution, which may be feasible or not. Effectiveness of the algorithm is examined using a medium size problem.

Problem Statement

The cost optimal sensor network of a system with precision constraints is obtained solving the following optimization problem[1]

$$\text{Min} \sum_{\forall i} c_i q_i$$

$$\text{s.t.}$$

$$\sigma_i(q) \le \sigma_i^* \quad \forall\, i \in M_S$$

$$q_i = 0,1 \quad \forall\, i \tag{1}$$

* To whom correspondence should be addressed. Tel.: 405 325-5458. Fax: 405 325-5813. E-mail: bagajewicz@ou.edu.

10.1021/ie050794w CCC: $33.50
Published on Web 08/25/2006

where q_i is the vector of binary variables indicating that a sensor measures variable i, c_i is the cost of such a sensor, and M_S represents the set of variables where a certain desired precision is required (variables of interest/key variables). The precision constraint ($\sigma_i(q) \leq \sigma_i^*$) states that the variance of the estimator, obtained through data reconciliation (or through some other means) has to be less than or equal to certain variance threshold $(\sigma_i^*)^2$ for the variables of interest (M_S).

It is well-known that networks obtained using only precision constraints usually perform in a poor manner when gross errors are present, and therefore, one needs to have some software redundancy to be able to filter these gross errors. Bagajewicz[1] showed one way to accomplish this proposing that three more properties be added: residual precision, resilience, and error detectability. Together, they can enable the data reconciliation/gross error detection procedure the right gross error filtering capabilities. We now briefly review these concepts.

Residual Precision. Residual precision is the ability of the network to guarantee a certain level of precision in key selected variables when gross errors are detected and the measurements are eliminated.[1] Formally, a variable has a residual precision of order k when the specified value of residual precision is maintained even after k gross errors, regardless of their position in the network, are detected and the measurements are eliminated.[1]

Error Detectability. The ability of the network to detect k gross errors of a certain adimensional size κ^D or larger is called error detectability of order k.[1] More specifically, when measurements follow a normal distribution, the objective function of data reconciliation follows a central chi-square distribution with m degrees of freedom. Moreover, a gross error of size δ_i in the variable x_i follows a noncentral chi-square distribution $\chi_m^2(\omega)$, where ω is the noncentrality parameter. Using these concepts (in turn developed by Madron[19,20]), Bagajewicz[1] provided an inequality that relates κ^D to the noncentrality parameter and the variances of the measurements and the estimator, respectively.

$$\kappa_i^D \geq \omega \frac{\sigma_{i,m}}{(\sigma_{i,m}^2 - \sigma_i^2)^{1/2}} \tag{2}$$

This inequality needs to hold for gross error detectability of order $k = 1$. No inequalities were developed for higher order.

Resilience. If a gross error of a certain magnitude occurs in some variable and is not detected, a certain corruption of data will take place when data reconciliation is performed. The ability of the network to limit the smearing effect of k undetected gross errors of a certain adimensional size or lower is called gross error resiliency of order k.[1]

The optimization model containing all these constraints can be written as follows[1,15]

$$\begin{aligned} &\text{Min} \sum_{\forall i} c_i q_i \\ &\text{s.t.} \\ &\sigma_i(q) \leq \sigma_i^* \quad \forall\, i \in M_S \\ &\psi_i(q,k_i) \leq \psi_i^*(k_i) \quad \forall\, i \in M_P \\ &\omega \frac{\sigma_{i,m}}{(\sigma_{i,m}^2 - \sigma_i^2)^{1/2}} \leq \kappa_D \quad \forall\, i \in M_D \\ &\tau_i(k,\kappa_R) \leq r_i^* \quad \forall\, i \in M_R \end{aligned} \tag{3}$$

where M_P, M_D, and M_R are set of variables where certain desired residual precision, error detectability, and resilience, respectively, is required, $\psi_i(q,k_i)$ and $\psi_i^*(k_i)$ represent the residual precision (function of q) of order k and its threshold, respectively, and $\tau_i(k,\kappa_R)$ and r_i^* represent the resilience level and the threshold, respectively. A detailed expression for $\sigma_i(q)$, $\psi_i(q,k_i)$, and $\tau_i(k,\kappa_R)$ can be found in the apper and book by Bagajewicz.[1,15]

All of the above is equivalent to requesting a certain level of software accuracy, a concept we briefly review here.

Software Accuracy. Stating that previously defined accuracy had little practical value, Bagajewicz[21] introduced a new definition of accuracy of estimator defined as the sum of the maximum undetected induced plus the precision of the estimator. Under the assumption of the use of maximum power test with confidence p to obtain the maximum undetected induced bias and presence of only one gross error in the system, the accuracy of the estimator of a variable i is given by

$$\underset{a_i}{\sim} \text{MP}(p,1) = \sqrt{\hat{S}_{ii}} + Z_{\text{crit}}^{(p)} \quad \underset{\forall s}{\text{Max}} \frac{[I - (SW)_{is}]}{\sqrt{W_{ss}}} \tag{4}$$

where $Z_{\text{crit}}^{(p)}$ is the critical value for the test at the confidence level p, S is the variance−covariance matrix of the measurements, and $W = A^T(ASA^T)^{-1}A$ in which A is the incidence matrix. More recently, Bagajewicz[22] showed a Monte Carlo approach to obtain an alternative definition of accuracy.

When using software accuracy, the optimization design model is

$$\begin{aligned} &\text{Min} \sum_{\forall i} c_i q_i \\ &\text{s.t. } a_i(q) \leq a_i^* \quad \forall\, i \in Is \end{aligned} \tag{5}$$

These problems are MINLP because all the variables involved in the constraints (precision, residual precision, resilience, and error detectability, as well as accuracy) are nonlinear functions of q. These functions have not been originally written in explicit form: the difficulty stems from the fact that the dimensions of the matrices involved in the calculation of the variance a function of q. In other words, the matrices are constructed only after q is known and observability and redundancy analysis can be done. To overcome this difficulty, Bagajewicz and Cabrera[2] as well as Chmielewski et al.[3,4] introduced the concept of fictitious sensors with very large variance: the former exhibited scaling problems, and the latter, who used linear matrix inequalities, did not solve large problems; so, the computational efficiency is not known.

The original tree type enumeration proposed by Bagajewicz,[1] which has some special pruning capabilities is inefficient for fairly large systems. For example, a 24 stream flowsheet problem (Figure 13) solved using the tree enumeration method using an Intel Celeron 1.39 GHz, 256 Mb RAM PC processor takes 8 h, 18 min, and 25 s to complete. Larger systems simply take unacceptably longer times.

We now focus on the design of sensor networks and present the basic elements of the new approach.

Connection between Cutsets and Observability/Redundancy

Process graphs have been extensively used in process flowsheeting and in other applications of process design. Each unit in a process flow diagram represents a node and the streams joining these units represent the edges of the graph. A hypothetical node called the environmental node is also included in the process graph to make the process graph cyclic (closed).

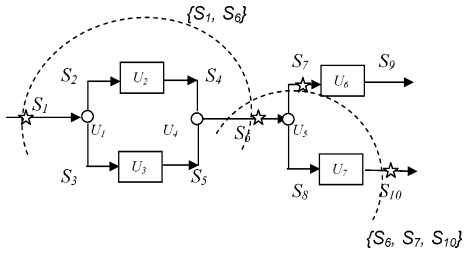

Figure 1. Two cutsets.

Important terms and concepts related to graph theory used in this article are defined in the Appendix.

Kretsovalis and Mah[23] pointed out that a cutset corresponds to a set of variables for which a material balance involving a certain number of units can be written. Thus, the number of all cutsets containing a variable S_i is equal to the number of material balances that can be written involving variable S_i. If all the streams forming a cutset in a process graph are measured, they all are redundant with a degree of redundancy of 1 with respect to the variable of interest. In other words, one can say that a measurement is redundant if it lies on at least one cutset in which all the variables are measured. In addition, by increasing the number of such cutsets containing the measurement of S_i, the degree of redundancy of that variable also increases. The concept of the degree of redundancy was first proposed by Luong et al.[24] and later generalized by Bagajewicz and Sánchez.[25]

Our purpose is then to replace the tree searching algorithm based on nodes of the tree built using a union of single measurements with a tree searching algorithm in which the nodes are a union of cutsets.

For example, consider the flow sheet of Figure 1, where streams S_1, S_6, S_7, and S_{10} are measured and consider that variable S_6 is of interest, that is, that precision and gross error robustness are required in this variable. These measurements are all redundant because they are part of the cutsets $\{S_1, S_6\}$ and $\{S_6, S_7, S_{10}\}$, as illustrated in Figure 1. More specifically, the key variable S_6, is redundant and has a degree of redundancy of 2.

Although in the majority of the cases one is interested in redundant networks, there are special cases where one may be forced to give up this requirement. This leads to a nonredundant network requiring only observability in some variables of interest. It is easy to see how cutsets of the process graph can be used to make variables of interest observable without requiring redundancy (or the ability to filter gross errors efficiently), even without measuring them. Indeed, in the above example, if stream S_6 is the variable of interest and is not measured but the rest are, it is observable because it lies on two cutsets $\{S_1, S_6\}$ and $(S_6, S_7, S_{10}\}$, where all the streams except S_6 are measured. This is because a material balance can be used to estimate S_6. There is no other way of making a nonmeasured variable observable.

We now explain the different types of cutsets present in the process graph related to a key variable and how each type contributes to the observability and redundancy of the key variable. Thus, all the cutsets of a process graph can be classified in to the following types (Figure 2).

(a) Cutsets containing one or more numbers of key variables.

(b) Cutsets containing one or more variables which are present in type a cutsets, i.e., intersecting the type a cutset at one or more locations (variables).

(c) Cutsets containing one or more numbers of variables which are present in type b or other type c cutsets, i.e., intersecting type b or other type c cutsets at one or more locations (variables).

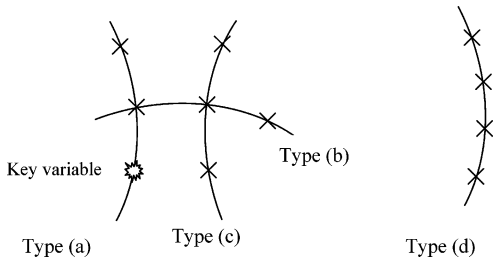

Figure 2. Types of cutsets in a process graph.

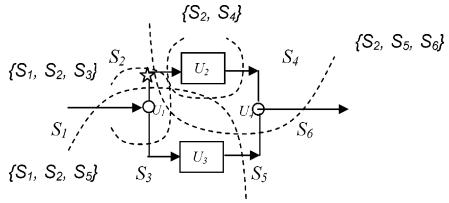

Figure 3. Type a cutsets corresponding to key variable S_2.

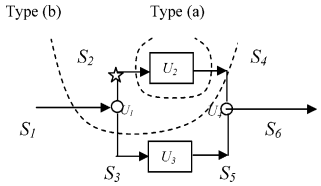

Figure 4. One type a cutset and one type b cutset.

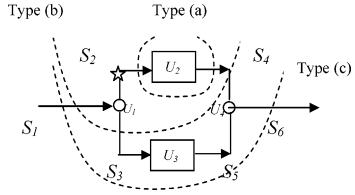

Figure 5. One type a cutset, one type b cutset, and one type c cutset.

(d) Cutsets not containing any variables present in type a, b, and c cutsets, i.e., not intersecting type a, b, or c cutsets.

Type a cutsets are the ones which directly provide redundancy in the key variable when all the variables in it are measured and provide observability when all variables except the key variable are measured.

Consider now a type b cutset. If the measurement in the intersection of this type b and the type a cutsets is lost, then, the rest of the measurements in the type b cutset provide the alternative means of estimating the lost measurement. In this way, the redundancy needed to calculate the key variable is not lost. We illustrate this now through an example.

Consider the system of four units and six streams with S_2 as the key variable (variable of interest) shown in Figure 3. The figure illustrates all the type a cutsets: $(\{S_1, S_2, S_3\}, \{S_2, S_4\}, \{S_2, S_5, S_6\}, \{S_1, S_2, S_5\})$. Estimability in the key variable is obtained either by measuring the key variable or measuring all the variables of any chosen cutset or set of cutsets (one can choose more than one). In turn, redundancy is achieved by measuring all the variables in the chosen cutsets.

Figure 4 illustrates the case of one type a cutset $\{S_2, S_4\}$ and one type b cutset $\{S_1, S_3, S_4\}$. Finally, Figure 5 illustrates the addition of a type c cutset $\{S_1, S_6\}$ to the type a and b cutsets of Figure 4.

Addition of a type b cutset to a type a cutset increases the estimability of the key variable. A type d cutset does not contain any variable of the type a, b, or c cutsets, and therefore, its measurements do not contribute in any way to the estimation of the key variable in the type a cutset; for this purpose, they

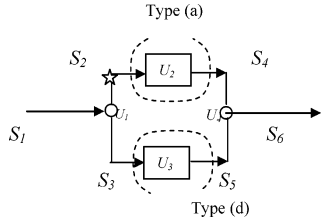

Figure 6. One type a cutset and one type d cutset.

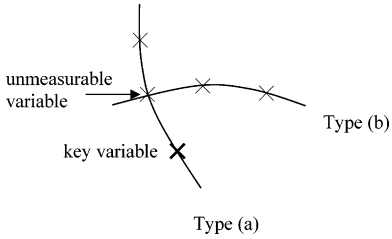

Figure 7. Type a cutset with an unmeasurable variable.

are useless and can be discarded. Figure 6 illustrates the addition of type d cutset $\{S_3, S_5\}$ to a type a cutset $\{S_2, S_4\}$.

In cases when it is not feasible to measure a variable of a type a cutset, type b cutsets intersecting that variable can be used to provide observability in the unmeasured variable, which in turn provides estimability of the key variable. Figure 7 shows such a case.

Thus, on the basis of the above concepts, when redundancy is required for a key variable (and by extension several key variables), a set of measurements corresponding to a certain set of cutsets is a feasible solution by simply making the union of the chosen cutsets after the following steps are taken: (a) disregarding any type d cutset chosen; (b) disregarding any type b cutset that contains unmeasurable variables that do not belong to type a cutsets; (b) disregarding any type a cutset that contains an unmeasurable variable that is not a key variable or does not belong to any other type b cutset.

One important property of any redundant feasible solution of the sensor location problem is that it can always be constructed using at least one combination of type a, b, and c cutsets, as it was illustrated above. This self-evident property can be easily proven by contradiction: Indeed, suppose not, that is, suppose that a feasible solution cannot be constructed using the union of cutsets. This means that the measurements used are not part of a type a cutset and, consequently, do not participate in a mass balance. This would mean that nonredundant measurements are redundant, which is a contradiction.

In addition if only estimability is required, then one can also consider adding using the measurement of the stream itself or using the aforementioned cutsets without measuring the key variable.

The exclusion of type d cutsets from consideration is correct for all the cases where two vertices (units in our case) are connected with only one edge (stream). However, for graphs in which two vertices (units in our case) are connected by one or more edges (streams in our case) and one of the vertices is not connected to any other vertices of the graph, type d cutsets contribute toward the estimability of the key variable. Consider, for example, the system of Figure 8, where we have unit 4 connected to unit 1 through two streams. All the cutsets of the system are $\{S_7, S_8\}$, $\{S_1, S_2\}$, $\{S_1, S_3\}$, $\{S_1, S_4, S_5, S_6\}$, $\{S_2, S_3\}$, $\{S_2, S_4, S_5, S_6\}$, and $\{S_3, S_4, S_5, S_6\}$. If we consider S_8 as the key variable, $\{S_7, S_8\}$ is a type a cutset and the rest are type d. It is however obvious that measuring all variables around unit 1 $\{S_7, S_8, S_1, S_2\}$ (union of $\{S_7, S_8\}$ and $\{S_1, S_2\}$) provides

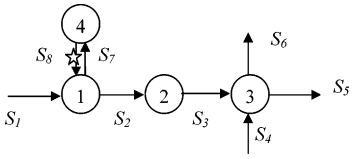

Figure 8. Special case system.

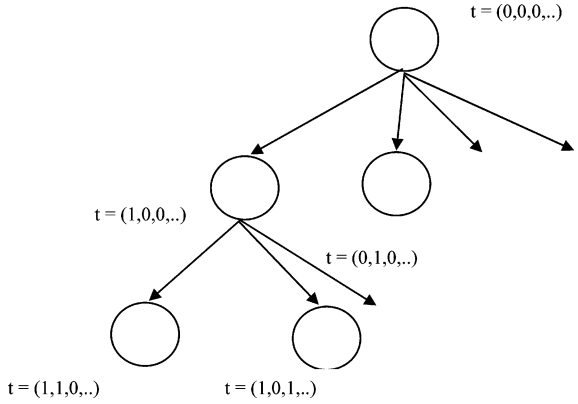

Figure 9. Tree of cutsets.

another source of redundancy. Hence, we do not discard type d cutsets in these special cases.

The above ideas can therefore be used in an implicit tree enumeration scheme, where each node is a set of cutsets chosen from the set of all cutsets of the system. On the basis of the above, we now propose a tree searching algorithm, similar to the one proposed by Bagajewicz,[1] but using cutsets as elementary network constructing units, instead of single measurements.

Implicit Tree Enumeration Algorithm

We propose a tree enumerating method with branching and bounding similar to the one developed by Bagajewicz[1] using a union of cutsets to define the nodes instead of a union of stream measurements (Figure 9). We use a vector $t = (c_1, c_2, ..., c_n)$ to represent a node, where c_i is a binary variable indicating that the cutset C_i is part of the solution. For the design of the redundant sensor network, we measure all the variables of the cutsets active in that node. However, for a nonredundant network (considering observability), at any node we measure all the variables of the active cutsets except the key variables. In addition, for the nonredundant network, we also consider measuring individual key variable as one of the building blocks to use in the tree along with other cutsets.

For example, all the cutsets of Figure 3 are $C_1 = \{S_1, S_2, S_3\}$, $C_2 = \{S_1, S_3, S_4\}$, $C_3 = \{S_1, S_2, S_5\}$, $C_4 = \{S_1, S_6\}$, $C_5 = \{S_2, S_4\}$, $C_6 = \{S_2, S_5, S_6\}$, $C_7 = \{S_3, S_5\}$, $C_8 = \{S_3, S_4, S_6\}$, and $C_9 = \{S_4, S_5, S_6\}$. For redundant sensor network design, we would consider all these cutsets for developing the proposed tree. Suppose now that one feasible solution is given by $t = \{110000000\}$, that is, cutsets C_1 and C_5 are active and that cutset C_1 was added first and cutset C_5 was selected next. Then, the sensor network is given by $C_1 \cup C_5 = \{S_1, S_2, S_3, S_4\}$ or $q = \{111100\}$. While for the design of a nonredundant sensor network we would again consider all the cutsets but without the key variable in it, we add the set $\{010000\}$ measuring key variable S_2, so now, all the building blocks for the tree would be $C_1 = \{S_1, S_3\}$, $C_2 = \{S_1, S_3, S_4\}$, $C_3 = \{S_1, S_5\}$, $C_4 = \{S_1, S_6\}$, $C_5 = \{S_4\}$, $C_6 = \{S_5, S_6\}$, $C_7 = \{S_3, S_5\}$, $C_8 = \{S_3, S_4, S_6\}$, $C_9 = \{S_4, S_5, S_6\}$, and $K_1 = \{S_2\}$. Suppose now that one solution to a nonredundant network is given by $t = \{000010000\}$, that is, cutset C_5 is active. Then, the sensor network is given by $C_5 = \{S_4\}$ or $q = \{000100\}$.

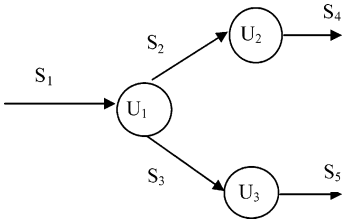

Figure 10. Example 1.

Table 1. Flow Rates for Example 1

streams	S_1	S_2	S_3	S_4	S_5
flow	150.1	52.3	97.8	52.3	97.8

Table 2. Cost of Flow Meters for Example 1

streams	S_1	S_2	S_3	S_4	S_5
cost	1500	2000	2500	1800	1700

The proposed procedure is as follows.

(1) Find all the cutsets of the process graph. In the literature, different algorithms are available to find all cutsets of a graph. We implemented the algorithm discussed in Bagajewicz.[1,15]

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

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

(4) Using the branch first rule, develop each branch by making one element of "*t*" active and adding one cutset at a time which is chosen from the remaining cutsets using a branching criterion.

(5) While performing the branching criteria, if any set of streams 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 a different set of cutsets.

(6) This is continued until the stopping criterion is met. In such a 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 cutset is chosen in such a way that the cost obtained by its union with the existing active cutset is minimum.

Stopping Criterion. Because adding a cutset always increases the cost, whenever a feasible node is found (one that satisfies all the constraint of the problem), the tree is not explored further down nor are any sister branches.

We illustrate the performance of the procedure with several examples.

Examples

Example 1. Consider the five stream problem with three units shown in Figure 10. We use this example just to illustrate the methodology not the computational speed. The flow rates of Table 1 were considered. It is assumed that the flow meters of 2% precision are available. Different costs were considered for different streams for illustrating the example; Table 2 was considered for the cost.

Precision is only required for variables S_3 with $\sigma_3^* = 2\%$. We first find all the cutsets of the flow diagram shown in Figure 10; all cutsets sorted in ascending order of cost are listed in Table 3. Now, we remove the key variables, if present, from the cutsets and again find the cost of the cutsets. In addition, an additional building block measuring the key variable S_3 is

Table 3. Sorted Cutsets for the Five Stream Problem

cutsets	streams	cost	cutsets	streams	cost
C_1	S_2, S_4	3800	C_4	S_1, S_2, S_5	5200
C_2	S_3, S_5	4200	C_5	S_1, S_3, S_4	5800
C_3	S_1, S_4, S_5	5000	C_6	S_1, S_2, S_3	6000

Table 4. Sorted Building Blocks for the Five Stream Problem

building blocks	streams	cost of
C_2	S_5	1700
K_1	S_3	2500
C_5	S_1, S_4	3300
C_6	S_1, S_2	3500
C_1	S_2, S_4	3800
C_3	S_1, S_4, S_5	5000
C_4	S_1, S_2, S_5	5200

added to the cutset list ($K_1 = \{S_3\}$). Now, all the building blocks (cutsets and measurements) are sorted in ascending order of cost (Table 4).

We start the cutset tree with a root node with no measurements added (trivially infeasible). Now, we start exploring the tree by adding the first building block C_2, and node $t = \{1000000\}$ is evaluated by adding the instruments on the streams S_5 ($q = \{00001\}$). The node is feasible; hence, we stop going down or even exploring the sister nodes; see Figure 11. The

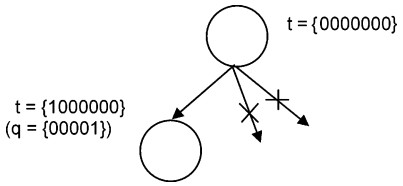

Figure 11. Tree of cutsets for example 1.

optimal solution is found at one node where cutset C_2 is considered without the key variable to give measurement in stream S_5 ($q = \{00001\}$) with a cost of 1700. The algorithm explores 1 node out of 127 nodes (total number of nodes $= 2^{\text{no of building blocks}} - 1$ or $2^7 - 1$). The algorithm took less than 1 s to solve this problem on an Intel Celeron 1.39 GHz, 256 Mb RAM PC. In comparison, using a tree of streams (as proposed by Bagajewicz[1]), 14 nodes are explored out of $2^5 - 1 = 31$ nodes and this also takes less than 1 s. All the times reported for this example and the ones that follow include preprocessing steps 1 and 2 of the algorithm (finding cutsets and sorting them).

The requirement of key variables to be observable gives a nonredundant network. Next, we design a redundant network for example 1 by adding residual precision. Now, precision is only required for variables S_1 and S_3 with $\sigma_1^* = 2\%$ and $\sigma_3^* = 2\%$. A residual precision of order 1 ($k = 1$) and a threshold of $\psi_1^* = 3\%$ and $\psi_3^* = 3\%$ are required. We again consider the sorted cutsets listed in Table 3.

We start the cutset tree with a root node with no cutsets added (trivially infeasible). Now, we start exploring the tree by adding the first cutset C_1, and node $t = \{100000\}$ is evaluated by adding the instruments on the streams S_2 and S_4 ($q = \{01010\}$) which form the cutset C_1. The node is infeasible; hence, we go to the next level in the tree, $t = \{110000\}$. Here, cutset C_1 is active and we select a new cutset to add from the remaining five cutsets using the branching criteria. We now illustrate the branching criteria. The costs of adding one of the five cutsets are $C_1 \cup C_2 = 8000$, $C_1 \cup C_3 = 7000$, $C_1 \cup C_4 = 7000$, $C_1 \cup C_5 = 7800$, and $C_1 \cup C_6 = 7800$. Thus, the selected cutset to add is C_4 and instruments are placed on streams S_1, S_2, S_4, and S_5 ($q = \{11011\}$). The node is infeasible; hence, we go deeper into the tree, selecting one more cutset to be added from the remaining four cutsets.

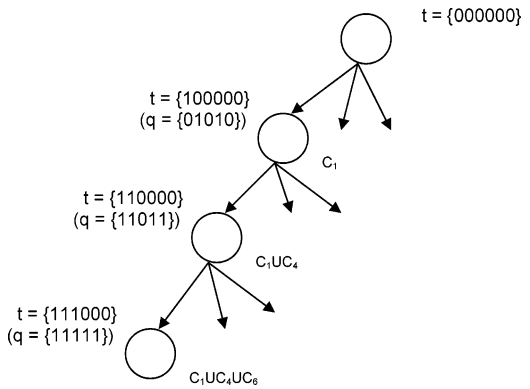

Figure 12. Tree of cutsets for example 1 (redundant network).

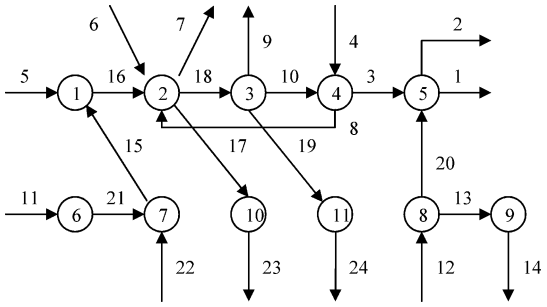

Figure 13. Example 2 (following the work of Madron and Veverka[10]).

Table 5. Cost of Flowmeters (Meyer et al.[24])

streams	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9
cost	19	17	13	12	25	10	7	6	5

At node $t = \{111000\}$, C_1 and C_4 are the active cutsets and we need to select and add one cutset from the remaining four cutsets, C_2, C_3, C_5, and C_6. The one that provides the lowest cost is C_3, but $C_1 \cup C_4 \cup C_3$ gives rise to $q = \{11011\}$, that is, streams S_1, S_2, S_4, and S_5, which was already considered in the previous node; hence, we discard it. The next choices are C_2, C_5, and C_6, which are all equivalent (have the same cost). Selecting C_6 gives rise to all streams being measured ($q = \{11111\}$).

We have found a feasible node, so one should explore sister nodes or move back up in the tree to explore all the branches, always using the branching criteria; see Figure 12. The optimal solution is found on node number 7 where a union of cutsets C_2 and C_5 is performed to give measurements in streams S_1, S_3, S_4, and S_5 ($q = \{10111\}$) with a cost of 7500. The algorithm explores a total of 11 nodes out of 63 nodes (the total number of nodes $= 2^{\text{no of cutsets}} - 1$ or $2^6 - 1$). The algorithm took less than 1 s to solve this problem on an Intel Celeron 1.39 GHz, 256 Mb RAM PC. In comparison, using a tree of streams (as proposed by Bagajewicz[1]), 30 nodes are explored out of $2^5 - 1 = 31$ nodes and this also takes less than 1 s.

Example 2: Retrofit Problem. This example shows another aspect of the algorithm, one that is related to the selection of cutsets in retrofit situations, not to the computational speed. The network shown in Figure 13 was proposed by Madron and Veverka.[10] It consists of 24 streams and 11 nodes, 9 of which are unmeasured (streams S_1-S_9), and 15 streams are already measured ($S_{10}-S_{24}$). The streams which are to be measured are streams S_1-S_9. Meyer et al.[26] added the costs of Table 5, and the flowrates of Table 6 added by Bagajewicz and Cabrera[2] were considered. It is assumed that the new candidate sensors and the existing sensors have a precision of 2.5%.

The streams S_1-S_5 were requested to be observable with a residual precision order of 1 and a threshold of 30% ($\psi_i^* \leq$ 30%, $i = 1, ..., 5$). The flowsheet of Figure 13 has a total of 116 cutsets. Of the entire list, some cutsets contain different already measured variables but the same potentially measured variables. Thus when a new cutset has this condition, it is discarded. For example, cutset $C_1 = \{S_1, S_2, S_3, S_{20}\}$ and cutset $C_2 = \{S_1, S_2, S_3, S_{12}, S_{13}\}$ have the same potentially measured variables (S_1, S_2, and S_3) and differ only in measured variables. Hence, only C_1 is included in the list of cutsets. By this procedure, the list of cutsets is reduced to only 20, which were used to develop the tree. The solution is $\{S_1, S_2, S_4, S_5, S_8\}$ with a total cost of 79. Results for this method and the implicit enumeration of streams[1] are shown in Table 7. The computational time of the tree algorithm also includes the preprocessing time (steps 1 and 2). Consider adding an error detectability of $\kappa^D = 3.9$ (with $\gamma = 50\%$) for flows 1 and 3, which are the larger flows. Bagajewicz[1] found the solution to be $\{S_1, S_2, S_4, S_5, S_6, S_7\}$ with a total cost of 90. Now consider the addition of error resilience: if error resilience is requested at a level of $\kappa^R = 3$ for streams 1 and 3 (S_1 and S_3), then the solution is again $\{S_1, S_2, S_4, S_5, S_6, S_7\}$ with total cost of 90.

Table 6. Flow Rates of Example 2

stream	flow	stream	flow	stream	flow
S_1	140	S_9	10	S_{17}	5
S_2	20	S_{10}	100	S_{18}	135
S_3	130	S_{11}	80	S_{19}	45
S_4	40	S_{12}	40	S_{20}	30
S_5	10	S_{13}	10	S_{21}	80
S_6	45	S_{14}	10	S_{22}	10
S_7	15	S_{15}	90	S_{23}	5
S_8	10	S_{16}	100	S_{24}	45

Table 7. Comparison with Stream Tree Enumeration (Example 2)

	implicit tree enumeration using cutsets		implicit tree enumeration method[1]	
	comput time[a]	nodes explored[b]	comput time[a]	nodes explored[c]
residual precision	<1 s	60	2 s	66
error detectability	<1 s	72	3 s	92
error detectability and resilience	<1 s	70	3 s	88

[a] On an Intel Celeron 1.39 GHz, 256 Mb RAM PC. [b]Total nodes: $2^{20} - 1$. [c]Total nodes: $2^9 - 1$.

Table 8. Cost of Flow Meters of Example 3

stream	cost	stream	cost	stream	cost
S_1	19	S_9	5	S_{17}	17
S_2	17	S_{10}	13	S_{18}	18
S_3	13	S_{11}	17	S_{19}	17
S_4	12	S_{12}	13	S_{20}	15
S_5	25	S_{13}	12	S_{21}	15
S_6	10	S_{14}	12	S_{22}	13
S_7	7	S_{15}	17	S_{23}	13
S_8	6	S_{16}	19	S_{24}	13

Example 3: Observability in a Grassroots Design Problem. Consider again the flowsheet in Figure 13 with the same flowrates but now none of the streams initially measured. All measurement candidates also have a precision of 2.5% but the costs of Table 8.

A precision of 2.5% is required for variables S_3, S_{10}, S_{16}, S_{17}, S_{20}, and S_{24} ($\sigma_i^* = 2.5\%$, $i = 3, 10, 16, 17, 20, 24$). The proposed cutset algorithm explored 937 nodes out of a total of $2^{154} - 1$ nodes and obtained the solution $\{S_3, S_{10}, S_{16}, S_{20}, S_{23}, S_{24}\}$ with a total cost of 86. The same problem was solved using the implicit tree type enumeration method proposed by Bagajewicz[1] and the same solution was found. Table 9 shows the comparison.

Example 4: Grassroots Redundant Design Problem. This example is considered to illustrate the power of the method in

Table 9. Comparison with Tree Enumeration (Example 3)

	implicit tree enumeration using cutsets	implicit tree enumeration method[1]
comput time	37 s	43 min 18 s
nodes explored	937 out of $2^{154}-1$	649 661 out of $2^{24}-1$

Table 10. Solution of Example 4

streams	cost
S_3, S_4, S_6, S_7, S_8, S_9, S_{10}, S_{12}, S_{13}, S_{16}, S_{17}, S_{19}, S_{20}, S_{23}, S_{24}	185
S_3, S_4, S_6, S_7, S_8, S_9, S_{10}, S_{12}, S_{14}, S_{16}, S_{17}, S_{19}, S_{20}, S_{23}, S_{24}	185

Table 11. Comparison with Stream Tree Enumeration (Example 4)

	implicit tree enumeration using cutsets	implicit tree enumeration method[1]
comput time	1 min 48 s	8 h 18 min 25 s
nodes explored	4 219 out of $2^{116}-1$	12 030 430 out of $2^{24}-1$

reducing computational time. Consider again the flowsheet in Figure 13 with the same flowrates and none of the streams initially measured. The cost of Table 8 was considered. A precision of 2.5% is required for variables S_3, S_{10}, S_{16}, S_{17}, S_{20}, and S_{24} ($\sigma_i^* = 2.5\%$, $i = 3, 10, 16, 17, 20, 24$). A residual precision of order of 1 and a threshold of 5% in the same variables ($\psi_i^* = 5\%$, $i = 3, 10, 16, 17, 20, 24$) are also required. The results obtained are shown in Table 10. The performance of the cutset algorithm is remarkably better and is shown in Table 11. Now, consider the addition of error detectability of $\kappa^D = 3.9$ (with $\gamma = 50\%$) for flows 3 and 10. Both the proposed algorithm and the implicit stream tree enumeration methods found the same solution exploring the same number of nodes.

Conclusions

A tree algorithm, based on graph theory and the use of cutsets, was developed to solve a cost-optimal sensor network problem. Cutsets proved to give solutions for any general (redundant and nonredundant) sensor networks. The developed cutset algorithm performs efficiently for medium size problems but does not work as expected for large size problems. This remains a challenge for future work.

Appendix

Review of Graph Theory[27]: Graph. A set of vertices connected by edges. Figure A1 shows a graph with five edges and four vertices.

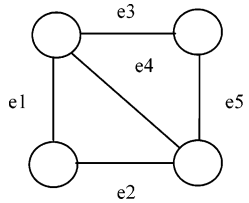

Figure A1. Graph with five edges and four vertices.

Cycle. A set of connected edges of a graph such that the last edge is connected to the first. Consider Figure A1, {*e*1, *e*2, *e*4} and {*e*3, *e*4, *e*5} forms the cycle.

Tree. A set of connected edges that does not form a cycle. Figure A2 shows a few trees of the graph shown in Figure A1. This should not be confused with the notion of tree search or tree enumeration that is used in the body of the paper, which refers to enumeration techniques used for optimal solution searches.

Figure A2. Trees of the graph of Figure A1.

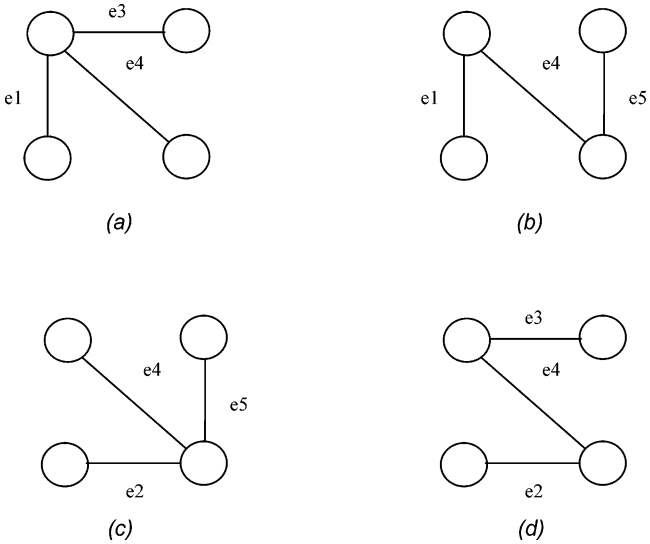

Figure A3. Spanning trees of the graph of Figure A1.

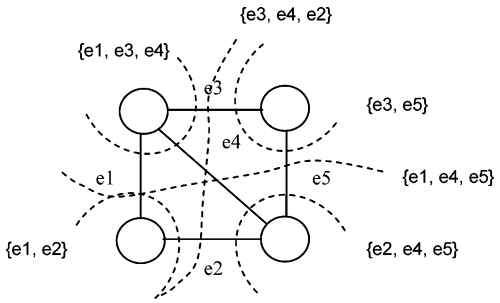

Figure A4. Cutsets of the graph of Figure A1.

Spanning Tree. A tree that connects all the vertices of the graph. Figure A3 shows all the spanning trees of the graph in Figure A1.

Cutset. The set of edges of a graph that, when eliminated, separates the graph into two disjoint subgraphs. The sets {*e*1, *e*3, *e*4}, {*e*1, *e*4, *e*5}, {*e*3, *e*5}, {*e*1, *e*2}, {*e*3, *e*4, *e*2}, and {*e*2, *e*4, *e*5} are cutsets of the graph shown in Figure A1. These are shown in Figure A4.

Fundamental Cutsets. A cutset of a graph which contains exactly one branch of a spanning tree. Referring to the spanning tree, Figure A3b, the fundamental cutsets are {*e*1, *e*2}, {*e*3, *e*4, *e*2}, and {*e*3, *e*5}. These are shown in Figure A5. All the cutsets of the graph can be obtained using fundamental cutsets.

Ring Sum of Cutsets. A ring sum operation defined as the union minus the intersection. For example, the ring sum of {*e*1, *e*2} and {*e*3, *e*4, *e*2} has as a result {*e*1, *e*3, *e*4}, which is the union of these two sets minus its intersection {*e*2}.

Property. Any cutset of the graph can be obtained by a ring sum of a set of fundamental cutsets.

The sets {*e*1, *e*3, *e*4}, {*e*1, *e*4, *e*5}, {*e*3, *e*5}, {*e*1, *e*2}, {*e*3, *e*4, *e*2}, and {*e*2, *e*4, *e*5} are cutsets of the graph shown in Figure

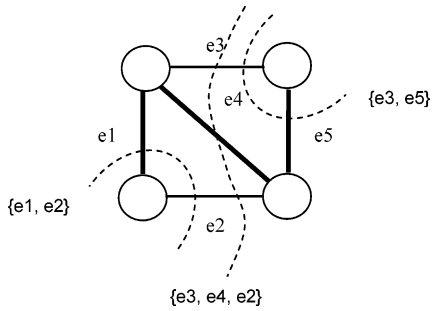

Figure A5. Fundamental cutsets of the graph of Figure A1.

A1. Referring to the spanning tree, Figure A3b, the fundamental cutsets are $\{e1, e2\}$, $\{e3, e4, e2\}$, and $\{e3, e5\}$; all the other cutsets are obtained by performing a ring sum on the fundamental cutsets.

$$\{e1, e3, e4\} = \{e1, e2\} \times \{e3, e4, e2\}$$

$$\{e1, e4, e5\} = \{e1, e2\} \times \{e3, e4, e2\} \times \{e3, e5\}$$

$$\{e2, e4, e5\} = \{e3, e4, e2\} \times \{e3, e5\}$$

Literature Cited

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

(2) Bagajewicz, M.; Cabrera. E. A New MILP Formulation for Instrumentation Network Design and Upgrade. *AIChE J.* **2001**, *48* (10), 2271 2001.

(3) Chmielewski, D.; Palmer, T. E.; Manousiouthakis, V. Cost Optimal Retrofit of Sensor Networks with Loss Estimation Accuracy. Presented at the AIChE Annual Meeting, Dallas, TX, November, 1999.

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

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

(6) Carnero, M.; Hernandez, J.; Sánchez, M.; Bandoni, A. An Evolutionary Approach for the Design of Nonredundant Sensor Networks. *Ind. Eng. Chem. Res.* **2001**, *40* (23), 5578.

(7) Carnero, M.; Hernandez, J.; Sánchez, M.; Bandoni, A. An Evolutionary Approach for the Design of Nonredundant Sensor Networks. *Ind. Eng. Chem. Res.* **2005**, *44* (2), 358.

(8) Vaclavek V.; Loucka., M. Selection of measurements necessary to achieve multicomponent mass balances in chemical plant. *Chem. Eng. Sci.* **1976**, *31*, 1199.

(9) Kretsovalis, A.; Mah, R. S. H. Observability and Redundancy Classification in Multicomponent Process Network. *AIChE J.* **1987**, *33*, 910.

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

(11) Ali, Y.; Narasimhan, S. Sensor Network Design for Maximizing Reliability of Linear Processes. *AIChE J.* **1993**, *39* (50, 820.

(12) Ali, Y.; Narasimhan, S. Redundant Sensor Network Design for Linear Processes. *AIChE J.* **1995**, *41* (10), 2237.

(13) Bagajewicz, M.; Sánchez, M. Design and Upgrade of Nonredundant and Redundant Linear Sensor Networks. *AIChE J.* **1999**, *45* (9), 1927.

(14) Bagajewicz, M.; Sánchez, M. Cost-Optimal Design of Reliable Sensor Networks. *Comput. Chem. Eng.* **2000**, *23*, 1757.

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

(16) Gerkens C.; Heyen, G. Sensor Network Design Using Genetic Algorithm. *Proceedings of the 11th IFAC Symposium on Automation in Mining, Mineral and Metal Processing*, Nancy, France, September 8−10, 2004.

(17) Gerkens C.; Heyen, G. Use of parallel Computers in Rational design of Redundant Sensor Networks. *Proceedings of the ESCAPE 14 Conference*, Lisbon; Portugal, Elsevier: New York, 2004.

(18) Carnero, M.; Hernández, J.; Sánchez, M. Optimal Sensor Network Design and Upgrade Using Tabu Search. *Proceedings of Escape 15*, Barcelona, Spain; Elsevier: New York, 2005.

(19) Madron, F. A New Approach to the Identification of Gross Errors in Chemical Engineering Measurements. *Chem. Eng. Sci.* **1985**, *40*, 1855.

(20) Madron F. *Process Plant Performance, Measurement Data Processing for Optimization and Retrofits*; Ellis Horwood: West Sussex, England, 1992.

(21) Bagajewicz, M. On the Definition of Software Accuracy in Redundant Measurement Systems. *AIChE J.* **2005**, *51* (4), 1201.

(22) Bagajewicz, M. On a New Definition of a Stochastic-based Accuracy Concept of Data Reconciliation-Based Estimators. *Proceedings of Escape 15*, Barcelona, Spain; Elsevier: New York, 2005.

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

(24) Luong, M.; Maquin, D.; Huynh, C.; Ragot, J. Observability, Redundancy, Reliability and Integrated Design of Measurement Systems. *Proceedings of the 2nd IFAC Symposium on Intelligent Components and Instrument Control Applications*, Budapest, Hungary, 1994.

(25) Bagajewicz, M.; Sánchez, M. Duality of Sensor Network Design Models for Parameter Estimation. *AIChE J.* **1999**, *45* (3), 661.

(26) Meyer, M. J.; Le Lann, L.; Koehret, B.; Enjalbert, M. Optimal Selection of Sensor Location on a Complex Plant Using Graph Oriented Approach. *Comput. Chem. Eng.* **1994**, *18* (Suppl.), S535.

(27) Deo, N. *Graph Theory with Application to Engineering and Computer Science*; Prentice-Hall: Englewood Cliffs, NJ, 2000.

Received for review July 5, 2005
Revised manuscript received July 8, 2006
Accepted July 11, 2006

IE050794W