Sensor Network Design for Maximizing Reliability of Linear Processes

Yaqoob Ali and Shankar Narasimhan
Dept. of Chemical Engineering, Indian Institute of Technology, Kanpur-208 016, UP, India

A new concept of reliability of estimation of variables is introduced which relates to the estimability of variables in the presence of sensor failures. Based on this concept, a method for optimal location of sensors in a pure flow process is developed. A graph-theoretic algorithm, SENNET, developed for this purpose, is shown to perform robustly and give globally optimum solutions for realistic processes.

Introduction

Chemical plants have grown larger in size and more integrated, requiring the use of computers for process data acquisition, monitoring, optimization and control. The efficiency and performance of these computers depend on extensive and accurate process data which are obtained through the measurements of such process variables as flow rate, temperature, concentration and level. Due to technical and economic feasibility, increased plant complexity due to additional instrumentation, and disturbances originating from the process or the environment, however, it is not possible to measure each and every process variable. Here, we exploit the mass and energy balance relationships between different variables of a steady-state process to estimate some or all of the unmeasured variables and more accurately estimate some measured variables. The estimability of variables depends on the topology of the process and the locations of sensors. The problem of selecting a set of variables to be measured, which is optimal with respect to some specified criteria, is called the sensor network design problem.

All variables that can be estimated either through their measurements or indirectly through their relationships with other measured variables are defined as observable. Moreover, if a measured variable can also be indirectly estimated by using measurements of other variables, then it is termed as redundant. For steady-state systems the concepts of observability and redundancy have been developed for pure mass flow (linear) processes (Mah et al., 1976) and for generalized processes including energy and multicomponent flows (Stanley and Mah, 1981a; Kretsovalis and Mah, 1987b, 1988; and Crowe, 1989). These concepts provide good qualitative criteria for sensor network design. Vaclavek and Loucka (1976) developed a strategy for sensor location in multicomponent networks so as to ensure the observability of all variables by assuming that either all compositions of a stream are measured or none at all. Kretsovalis and Mah (1987a) analyzed the effect of sensor location on the accuracy of estimation. Their approach was concerned essentially with the optimal allocation of redundant sensors, because they assumed that all variables are observable.

The problem of sensor location to ensure observability of dynamic systems has also been addressed before (Omatu and Seinfeld, 1989). In this article, however, we are concerned with steady-state processes only. The fundamental difference is that for steady-state systems only the solvability of a set of algebraic equations for a specified sensor location has to be checked, whereas in dynamic systems the entire transient system state during a time period has to be estimated, by making use of all measurements during that time period.

In this article, we introduce a new concept called the reliability of estimation of a variable. The reliability of estimation of a variable is simply the probability with which it can be estimated when sensors are likely to fail. This concept subsumes the concepts of observability and redundancy and also accounts for the probability of sensor failures. Using this concept, we develop a systematic strategy for designing an optimum sensor network. In addition, in our work we address the following questions.

1. How many different ways can a variable be estimated, and how should this aspect be taken into account in sensor network design?
2. Every sensor is prone to failure with a finite probability. If some of the sensors fail, can a variable still be estimated?
3. How do we design a sensor network to maximize the probability with which a variable can be estimated?

In this article, we deal only with a steady-state mass-flow

Correspondence concerning this article should be addressed to S. Narasimhan.

process. Moreover, we design the sensor network for the minimum number of sensors necessary to make all variables observable. We show that even in this case many different sensor networks can be designed, which ensure the observability of all variables. However, they are not all equivalent, if we take into account the probability of sensor failures. Interestingly, all networks are not equivalent even if all sensors have the same probability of failure.

Concept of Reliability of Estimation

First, we introduce the concept of reliability of estimating a variable, when sensors are likely to fail independently with known probabilities of failure. The reliability of estimating a variable is defined as the probability of estimating the variable for a given sensor network and for given probabilities of failure of the sensors. This concept is applicable to both measured and unmeasured variables.

Example 1

As an example, consider a simple process unit with one feed stream and two product streams, as shown in Figure 1. The mass flows of all streams are measured using sensors whose failure probabilities are all assumed to be 0.1. For this simple unit, the three flow variables are related to each other through a mass balance:

$$F_1 = F_2 + F_3$$

Assuming all sensor failures to occur independently and randomly, we obtain the reliability $R(F_1)$ of estimating F_1 as:

$$\begin{aligned} R(F_1) &= Pr\{S_1 \text{ is working}\} \text{ OR } Pr\ \{S_2 \text{ and } S_3 \text{ are working}\} \\ &= 0.9 + 0.81 - 0.9 \times 0.81 \\ &= 0.981 \end{aligned}$$

Note that the reliability of estimating F_1 is greater than the nonfailure probability of the sensor measuring F_1. This is due to the redundancy of F_1.

The reliability of estimating a variable encompasses the concepts of observability and redundancy as given by the following properties:

1. A variable is observable, if and only if the reliability of estimating a variable is greater than zero.
2. A measured variable is redundant, if and only if the reliability of estimating that variable is greater than the nonfailure probability of sensor measuring that variable.

In addition, the concept we propose accounts for the number of different ways in which a variable can be estimated as well as the sensor failure probabilities as demonstrated by example 1. The reliability of estimation of a variable is simply referred to as the reliability in the subsequent development.

Cutsets vs. Reliability of Estimation of Process Variables

In this section, we show how the cutsets of a process graph can be used to evaluate the reliabilities of process variables. Since the subsequent development makes extensive use of graph theory, some of the key terms and concepts are defined in

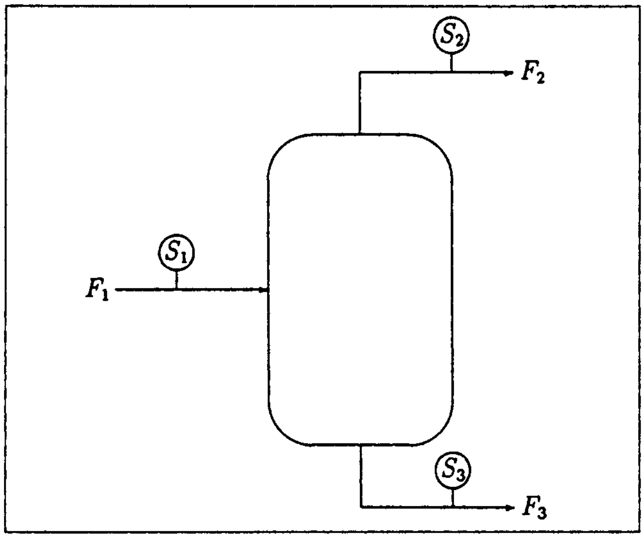

Figure 1. A simple process unit.

Appendix A. The description given in the Appendix is essentially drawn from Deo (1974).

The representation of a process by a process graph has been extensively used in process flowsheeting and in other applications of process design and analysis. The process units are modeled as nodes, and streams incident on these units are represented as edges. The process graph also contains a hypothetical node, called the environment node, from which it receives its feeds and to which it supplies its products. With the inclusion of the environment node, the process graph becomes cyclic. In general, each process stream may consist of several variables such as temperature, composition, and flow rate. Since we consider only total mass flows, each edge represents a unique mass-flow variable. Hence, in this development no distinction is made between the edge and the mass flow variable it represents. It should also be noted that the directions of edges are not important for the subsequent development, although in all our examples the directions of edges are shown for the sake of clarity.

Given a set of process variables measured, we first obtain the different possible ways through which each variable can be estimated using mass balance equations and measurements of other variables. The concept of cutsets is used for this purpose.

Cutsets have been used extensively in observability and redundancy classification (Kretsovalis and Mah, 1987b, 1988). Mah et al. (1976) have proved that for a pure mass-flow network, all variables are observable if the unmeasured variables form a spanning tree. It is also known (Deo, 1974) that a spanning tree of n nodes has $n-1$ edges (variables). From these two results, it immediately follows that the minimum number of sensors n_s required to make all variables observable is given by:

$$n_s = e - n + 1 \tag{1}$$

where e is the total number of edges.

The above result not only gives the minimum number of sensors required to make all mass flows observable but also tells us about their feasible locations. The strategy is to choose any spanning tree of the process graph and locate the sensors on the chords of the spanning tree. Thus, the chords and the branches of the spanning tree represent the measured and unmeasured streams, respectively.

A further observation is that, in the above case, there is a unique way for estimating every variable. While each measured variable can be estimated only through its measurement, each of the unmeasured variable can be estimated only through a fundamental cutset which consists of that variable (branch of the spanning tree) and some or all of the measured variables (chords of the spanning tree).

The reliability of each variable for minimum number of sensors can easily be evaluated. For a measured variable, the reliability is simply equal to the probability that the sensor measuring that variable does not fail. The reliability of estimating an unmeasured variable, i, is the probability that all sensors in the fundamental cutset containing that variable are in working state. It is given by:

$$R(i) = \prod_{\substack{j \in K_i^f \\ j \neq i}} (1 - p_j) \tag{2}$$

where K_i^f is the fundamental cutset containing variable i, and p_j's are the failure probabilities of the sensors corresponding to the chords in that fundamental cutset. Thus, the evaluation of reliabilities requires a procedure for obtaining all fundamental cutsets of the spanning tree (corresponding to the sensor network). We have implemented a straightforward algorithm for this purpose.

Sensor Network Design

If the requirement is simply that all variables should be observable, then any spanning tree of the process graph can be constructed and the sensors may be placed on the chords of the spanning tree. With respect to observability, all spanning trees are equivalent. However, different spanning trees lead to different reliabilities as shown by the following example.

Example 2

Consider the simplified ammonia network (Kretsovalis and Mah, 1988) which consists of six nodes and eight edges with node 6 representing the environmental node, as shown in Figure 2. Using Eq. 1, the minimum number of sensors required for this process is three, since $e-n+1=8-6+1=3$. Let us assume that sensors can be placed on any stream and each of them has a failure probability of 0.1.

Case 1. Mass flows in streams 1, 4 and 7 are measured. The mass flow of stream 6 can be estimated using the fundamental cutset (1, 4, 6, 7), which gives a reliability of 0.729.

Case 2. Mass flows in streams 4, 5 and 7 are measured. The mass flow of stream 6 is now estimated through the fundamental cutset (5, 6, 7) giving a reliability of 0.81, which is higher than that for case 1. The above example shows that a trade-off exists between sensor placement and reliability which can be utilized to design a sensor network.

Objective function

Clearly, the objective of a sensor network design can be maximizing the reliabilities of all variables. This, however, is

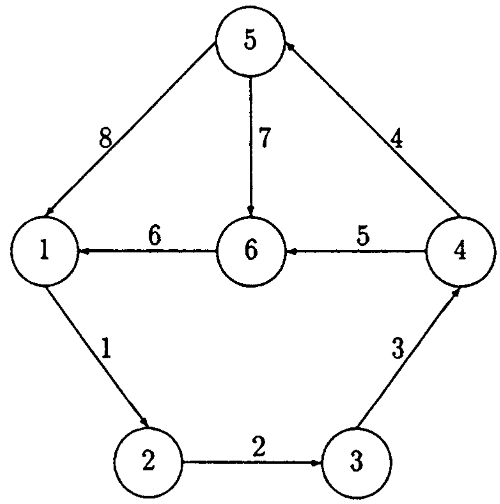

Figure 2. Simplified ammonia network.

not possible since only a minimum of sensors can be used. An attempt to maximize the reliability of any one particular variable may even lead to some other variables being unobservable. The objective we propose is to design a sensor network such that the minimum reliability among all variables is maximized. This objective is chosen based on the philosophy that a chain is no stronger than its weakest link. The integrated nature of the process leads to our logic that the reliability of the variable, which is the least, should be maximized by the proper selection of sensor locations. The minimum reliability of a sensor network is referred to as the network reliability.

Solution strategy

In a network, consisting of n nodes and e edges, the minimum number of sensors required is shown to be equal to $e-n+1$. If explicit enumeration is used, then $e!/(e-n+1)!(n-1)!$ combinations have to be examined to find the one that leads to maximum network reliability. In this process, however, many useless solutions which lead to unobservable variables are examined.

Alternatively, only those solutions that ensure observability of all variables can be explicitly generated. As already shown, these solutions correspond to the chords of spanning trees. Thus, all spanning trees of the process graph can be generated, and the one that leads to the maximum network reliability can be chosen. Many algorithms are available for generating all spanning trees of a graph (Aho et al., 1974; Deo, 1974; Nijenhuis and Wilf, 1978). Even in this case, however, the number of spanning trees can be fairly large. In the worst case the number of spanning trees can be as large as n^{n-2} (Deo, 1974). Thus when the number of units exceed eight, this approach cannot practically be used.

We propose an iterative improvement algorithm, which may not in all cases give the optimum solution but is about three to four orders of magnitude faster than explicit enumeration of all spanning trees. The iterative algorithm starts with an arbitrary spanning tree and at each iteration generates a new spanning tree with an improved network reliability. We make use of the following lemmas in the development of this iterative algorithm. The proofs of these lemmas are given in Appendix B.

Lemma 1. If a sensor is placed on one of the branches, say b_i, of a spanning tree, then one of the sensors placed on the chords belonging to the fundamental cutset K_i^f has to be removed. This will ensure that the new set of unmeasured variables form a spanning tree.

Remarks. This lemma describes a procedure for generating another spanning tree through the addition of a chord and deletion of an appropriate branch from a given spanning tree. This process is known as the elementary tree transformation (Deo, 1974).

Lemma 2. The ring sum of two fundamental cutsets, which have at least one common chord, gives a cutset of the graph.

Remarks. A ring sum of two fundamental cutsets may either give another cutset of a graph or a union of edge-disjoint cutsets. But, when the fundamental cutsets have a common chord, then we prove that their ring sum always gives a cutset of the graph. This property is useful in generating the fundamental cutsets of a new spanning tree obtained through an elementary tree transformation.

Lemma 3. For any sensor network the minimum reliability is always attained for an unmeasured variable and not for a measured variable.

Remarks. This lemma shows that in order to improve the network reliability of a given sensor network, we need to improve the reliability of some unmeasured variable.

Lemma 4. Let T be a spanning tree solution with branch b_x having the minimum reliability. Let K be the ring sum of K_x^f and K_q^f, where K_q^f is the fundamental cutset with respect to some branch b_q. If the failure probabilities of all sensors are equal, then the network reliability can be improved by placing a sensor on branch b_q and removing the sensor from chord c_p, provided the following conditions hold:

1. $c_p \epsilon K_q^f$ and $c_p \epsilon K_x^f$
2. $|K| < |K_x^f|$
3. If $c_p \epsilon K_j^f$ for any fundamental cutset, then $|K_j^f \oplus K_q^f| < |K_x^f|$

The variable b_q is denoted as the leaving variable and the variable c_p as the entering variable.

Remarks. The second condition listed above ensures that the reliability of variable b_x increases, while the third condition ensures that the reliabilities of all other unmeasured variables remain greater than the current network reliability.

There are three points to be noted with respect to Lemma 4. First, the network reliability cannot be improved by placing a sensor on variable b_x. This is because to maintain a spanning tree solution, the sensor of some chord of K_x^f should be removed (Lemma 1). The reliability of this variable in the new solution will then be equal to $R(b_x)$, and thus the network reliability is unchanged.

Secondly, if sensor failure probabilities are unequal, then the entering and leaving variables are still chosen in a similar manner except that instead of checking for the cardinality conditions 2 and 3 we explicitly evaluate the reliabilities of variables b_x and b_j's. Equation 2 can be used to evaluate these reliabilities.

Lastly, the converse of Lemma 4 is not true, that is, if we cannot find a branch b_q and chord c_p satisfying the three conditions, it does not imply that the global optimum solution has been obtained (although we can view it as a local optimum). Moreover, if conditions 2 and 3 do not hold as strict inequalities or if there are two or more variables with minimum reliability, then we get a degenerate solution (that is, the network reliability remains the same for the next solution and may continue for subsequent solutions as well). Both these problems are handled using heuristic strategies. The algorithm we propose is described below.

Algorithm—Equal Sensor Failure Probabilities

Based on the four lemmas, we develop an algorithm called SENNET for the SENsor NETwork design problem. For clarity of description, we consider the case when all the sensors have the same failure probabilities. The algorithm has some similarities with the SIMPLEX algorithm used in linear programming.

We start with an initial spanning tree and attempt to improve the network reliability by choosing a branch in which to place a sensor (leaving variable) and a chord from which to remove a sensor (entering variable). The algorithm is as follows:

Step 1. Generate a spanning tree of the process graph.

Step 2. Generate all the fundamental cutsets K_i^f of the spanning tree.

Step 3. Obtain K_{max} the set of fundamental cutsets that have the maximum cardinality. The branches corresponding to these fundamental cutsets are the variables with minimum reliability (compare with Eq. 2).

Step 4. Choose any element of K_{max} say K_x^f, which has not been considered before. Mark K_x^f as examined and go to step 5. If no such fundamental cutset exists go to step 10.

Step 5. Choose any other fundamental cutset K_q^f which has not been examined before. Obtain K, the ring sum of K_x^f and K_q^f. Mark K_q^f as examined and go to step 6. If no such K_q^f exists go to step 4.

Step 6. If $|K| \le |K_x^f|$ go to step 7. Otherwise, go to step 5.

Step 7. Compute the set $I = K_x^f - K$. Choose a chord c_p from set I which has not been examined before. Mark c_p as examined and go to step 8. If no such chord exists, go to step 5.

Step 8. For all fundamental cutsets K_j^f containing chord c_p, if $|K_j^f \oplus K_q^f| \le |K_x^f|$, go to step 9. Otherwise, go to step 7.

Step 9. Branch b_q is selected as a leaving variable and chord c_p is chosen to enter the spanning tree. Update all fundamental cutsets (to obtain the fundamental cutsets of the new tree) and go to step 3.

Step 10. Stop, if this step has already been executed ten times. Otherwise, store the current solution. Let branch b_q be selected as the leaving variable where b_q corresponds to the most recently examined cutset K_q^f and c_p is any arbitrary element of set $K_x^f - K$, where K is the ring sum of K_x^f and K_q^f. Go to step 3.

The optimum sensor network design corresponds to locating sensors on the chords of the "optimum" spanning tree. To understand the algorithm easily, a flow chart is given in Figure 3.

Handling degeneracy

In the above algorithm steps 6, 7 and 8, check to see that the conditions of Lemma 4 are satisfied. Note that in the algorithm the inequalities are weak inequalities and may lead to successive solutions which do not improve the network reliability (degeneracy). In fact, it is theoretically possible to be caught in an infinite loop. We attempt to break degeneracy by random selection of set K_x^f from set K_{max} in step 4 and random selection of chords c_p from set I in step 7. This heuristic method has worked well with the problems we tested.

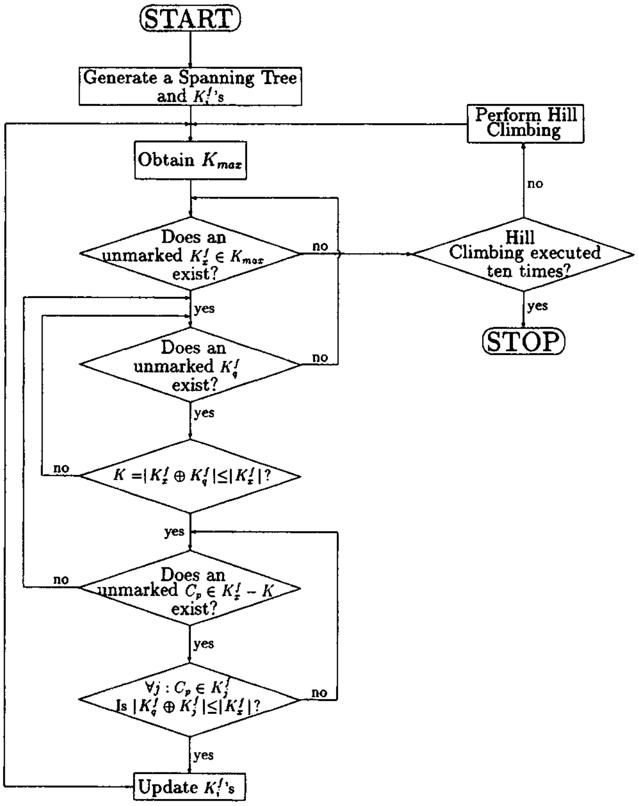

Figure 3. Flowchart of algorithm SENNET.

Hill climbing procedure

Step 10 in the above algorithm is implemented as an attempt to get away from a local minimum and reach the global optimum solution. If we reach a situation in which any choice of entering/leaving variable worsens the network reliability, then we perturb the current best solution and allow the network reliability to decrease and repeat the algorithm. This is akin to "hill climbing" strategies used in optimization. If after ten such attempts we are unable to improve the network reliability, then we choose the best current solution as optimum.

Updating the fundamental cutsets

In step 9, the fundamental cutsets of the new spanning tree can simply be obtained by updating the fundamental cutsets of the previous spanning tree as follows.

The fundamental cutset of the entering variable c_p is given by K_q^f. If a fundamental cutset K_j^f contains c_p as a member, then the updated fundamental cutset is given by the ring sum of K_j^f and K_q^f. All other fundamental cutsets remain unaltered.

Example 3

We illustrate our algorithm for the sensor network design of a simplified ammonia plant shown in Figure 2. As shown in Example 2, the minimum number of sensors required for this plant is three. Let the sensor failure probabilities for all edges be 0.1.

Step 1. Let the initial spanning tree solution consist of the unmeasured edges 2, 3, 5, 6 and 8.

Step 2. The fundamental cutsets are as follows (branches have an underscore):

(i) $(4, 7, \underline{8})$
(ii) $(1, 4, \underline{6}, 7)$
(iii) $(1, 4, \underline{5})$
(iv) $(1, \underline{2})$
(v) $(1, \underline{3})$

Steps 3 and 4. The set $K_{\max}$ consists of the maximum cardinality cutset ii in which variable 6 is observed using three sensor signals.This is the variable that has the minimum reliability and we attempt to improve its reliability. Thus,

$K_x^f = (1, 4, \underline{6}, 7)$

Steps 5 and 6. The ring sum of K_x^f with other fundamental cutsets gives the following cutsets:

(i) $(1, \underline{6}, \underline{8})$
(ii) $(\underline{5}, \underline{6}, 7)$
(iii) $(\underline{2}, 4, \underline{6}, 7)$
(iv) $(\underline{3}, 4, \underline{6}, 7)$

Among the above cutsets, only the first two have cardinality less than that of K_x^f. We arbitrarily choose the first. Thus, $K_q^f = (4, 7, \underline{8})$ and $K = (1, \underline{6}, \underline{8})$

Steps 7 and 8. The set I is obtained as $I = K_x^f - K = (4, 7)$. Any of the chords in set I may be chosen. We choose chord 4. Thus, $c_p = 4$.

Step 9. The branch $b_q = \underline{8}$ leaves the tree and chord 4 enters it. The new spanning tree obtained through the elementary tree transformation is (2, 3, 4, 5, 6). The following fundamental cutsets for this new spanning tree may be obtained by updating the previous solution (using Lemma 2).

(i) $(\underline{4}, 7, 8)$
(ii) $(1, \underline{6}, 8)$
(iii) $(1, \underline{5}, 7, 8)$
(iv) $(1, \underline{2})$
(v) $(1, \underline{3})$

Note that the reliability of variable $\underline{6}$ has improved though the network reliability remains the same, since cutset iii above contains three chords. This is due to the fact that for this cutset, condition 3 of Lemma 4 holds only as a weak inequality.

We can proceed with the next iteration in which 6 is the leaving variable and 1 is the entering variable to give the spanning tree solution (1, 2, 3, 4, 5). In the subsequent iteration, there is no choice of entering variables and a local optimum solution is obtained. We check if better solutions can be obtained by applying the hill climbing technique. However, in this case, the solution is not improved in ten successive attempts and we adopt the solution (1, 2, 3, 4, 5) as optimal. This solution corresponds to sensors on variables 6, 7 and 8.

To compare this solution with the global optimum, we implemented an algorithm (Nijenhuis and Wilf, 1978) for explicit enumeration of all spanning trees and obtained the best solutions. The results of this analysis are presented in Table 1. This table shows that the process graph has 32 spanning trees out of which only eight are globally optimal giving a network reliability of 0.81. It also shows the optimal solutions generated by our algorithm for 5 different initial starting solutions. All the solutions given by our algorithm are also found to be globally optimal, thus indicating that the algorithm is robust with respect to the choice of initial solutions and gives globally optimal solutions.

Table 1. Data and Results of Ammonia Plant

Data
No. of nodes: 6
No. of edges: 8
Failure probability of each sensor: 0.1

Results
Minimum no. of sensors: 3
No. of spanning trees: 32
No. of optimal solutions: 8
Network reliability: 0.810

Initial Solution	*Optimal Solution*
2 3 5 7 8	1 2 3 4 5
2 3 5 6 7	2 3 4 7 8
2 3 4 6 7	1 2 4 7 8
1 3 4 5 6	1 2 4 7 8
1 2 3 5 8	1 2 3 6 8

Table 2. Data and Results of Steam Metering Network (Equal Failure Probabilities)

Data
No. of nodes: 12
No. of edges: 28
Failure probability of each sensor: 0.10

Results
Minimum no. of sensors: 17
No. of spanning trees: 1,243,845
No. of optimal solutions: 125
Network reliability: 0.530

Initial Solution	*Optimal Solution*
4 9 10 12 13 17 20 24 25 26 28	2 4 9 10 12 17 20 24 25 26 28
4 9 10 12 13 17 18 21 22 26 27	1 2 4 9 12 17 21 23 25 27 28
1 2 3 4 5 10 14 15 16 26 28	1 2 4 8 9 10 17 21 22 24 27
1 2 9 10 12 17 18 20 22 27 28	1 2 4 9 10 17 21 22 25 27 28
2 4 5 8 12 15 21 23 25 26 28	1 2 4 8 10 17 21 22 24 27 28

Example 4

As a second example, consider the sensor network design for a steam metering network of a methanol plant (Serth and Heenan, 1986). This system is practically of reasonable size consisting of 12 nodes and 28 edges as shown in Figure 4. The environmental node (node 12) is not shown in the figure to maintain clarity. We have assumed that all sensors have a failure probability of 0.1.

The complete analysis of this network is presented in Table 2. In this case, the total number of spanning tree solutions is large, but only about 0.01% of these solutions is globally optimal giving a network reliability of 0.53. The worst sensor placement for this network corresponds to the initial spanning tree (1 2 9 10 12 17 18 20 22 27 28) which gives a network reliability of 0.17 (variable 28). The solutions generated by our algorithm for five different initial spanning trees are shown in Table 2. All these solutions are found to be globally optimal including the case when we start with the worst initial spanning tree solution (row 4 in Table 2). Although the algorithm is not guaranteed to give globally optimal solutions, at least for the several cases we tried, we did not obtain a suboptimal solution. The algorithm takes about 15 seconds on an IBM compatible PC using an 80386 processor and an 80387 coprocessor. In contrast, explicit enumeration of all spanning trees for this example requires about 17 hours of computing time. This again demonstrates the robustness and efficiency of the algorithm.

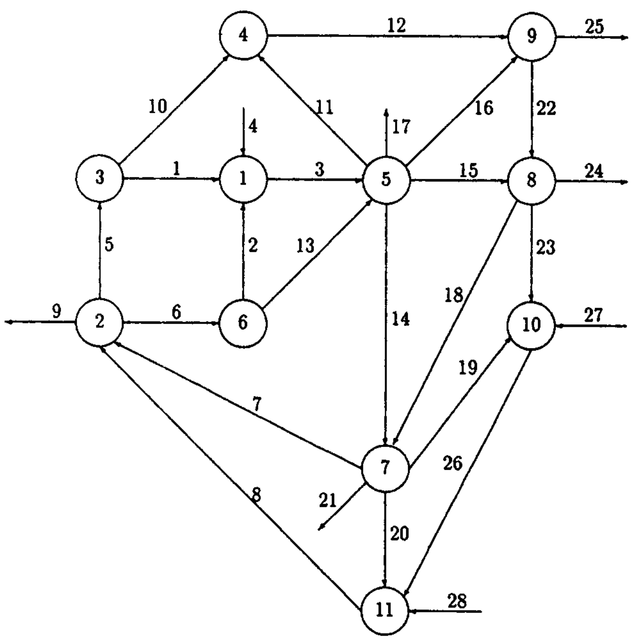

Figure 4. Steam metering network.

Unequal sensor failure probabilities

Finally, we examine the performance of the algorithm when sensor failure probabilities are unequal. The modifications to the algorithm for treating unequal sensor failure probabilities have been explained earlier. We again consider the steam metering system. The failure probabilities of the sensors are generated using random numbers. Two different sets of failure probabilities were generated and the algorithm was applied. The results in Tables 3 and 4 show that the number of optimal solutions has further decreased. Despite this, the algorithm could obtain the globally optimal solutions starting from each

Table 3. Data and Results of Steam Metering Network (Unequal Failure Probabilities)

Data
No. of nodes: 12
No. of edges: 28
Failure probabilities of sensors:
0.256 0.169 0.150 0.242 0.150 0.381 0.350 0.381 0.090 0.355 0.201
0.123 0.276 0.388 0.248 0.100 0.199 0.089 0.075 0.065 0.204 0.289
0.330 0.382 0.184 0.128 0.103 0.400

Results
Minimum no. of sensors: 17
No. of spanning trees: 1,243,845
No. of optimal solutions: 23
Network reliability: 0.204

Initial Solution	*Optimal Solution*
4 9 10 12 13 17 20 24 25 26 28	2 4 8 10 12 17 21 23 24 25 28
4 9 10 12 13 17 18 21 22 26 27	1 4 6 8 10 17 21 22 23 24 28
1 2 3 4 5 10 14 15 16 26 28	1 4 6 8 10 17 21 22 23 24 28
1 2 9 10 12 17 18 20 22 27 28	1 4 6 8 12 17 21 23 24 25 28
2 4 5 8 12 15 21 23 25 26 28	1 4 6 8 10 12 17 21 23 24 28

Table 4. Data and Results of Steam Metering Network (Unequal Failure Probabilities)

Data
Total number of nodes: 12
Total number of edges: 28
Failure probabilities of sensors:
0.266 0.143 0.111 0.078 0.344 0.125 0.223 0.138 0.293 0.346 0.315
0.137 0.200 0.177 0.145 0.057 0.231 0.144 0.239 0.322 0.123 0.102
0.325 0.201 0.057 0.256 0.275 0.125

Results
Minimum no. of sensors: 17
No. of spanning trees: 1,243,845
No. of optimal solutions: 92
Network reliability: 0.284

Initial Solution	*Optimal Solution*
4 9 10 12 13 17 20 24 25 26 28	1 4 6 9 12 17 21 23 25 27 28
4 9 10 12 13 17 18 21 22 26 27	1 4 6 9 10 17 21 24 25 27 28
1 2 3 4 5 10 14 15 16 26 28	1 4 6 9 10 17 21 22 25 27 28
1 2 9 10 12 17 18 20 22 27 28	1 4 6 9 12 17 21 24 25 27 28
2 4 5 8 12 15 21 23 25 26 28	1 4 6 9 10 17 21 24 25 27 28

of the five different spanning trees. It was observed that for this case, more iterations were required before the optimal solution was achieved. Furthermore, "hill climbing" technique proved more useful. On an average, the hill climbing technique had to be applied 5-6 times to obtain the global optimum.

Concluding Remarks

The problem of sensor location has been addressed in a complete process plant based on an entirely new and powerful concept of reliability of estimation of variables. The concept of reliability inherently contains the concepts of observability and redundancy and also accounts for sensor failures. A robust and efficient iterative improvement algorithm has been developed for sensor network design when the minimum number of sensors have to be installed in a pure mass-flow process. The extension to sensor network design for general processes requires further development. A comprehensive strategy for sensor network design that considers reliability, accuracy, and controllability needs to be developed, and the present work can serve as a starting point.

Notation

b_i = branch i of the spanning tree T
b_q = branch leaving the spanning tree T
b_x = branch with minimum reliability
c_i = chord i of the spanning tree T
c_p = chord entering the spanning tree T
e = number of edges in the process graph
$E(E')$ = set of edges of graph $G(G')$
F_i = flow rate in stream i
G = graph
G' = subgraph of G
I = set of entering variables defined in step 7 of algorithm
K = ring sum of K_x^f and K_q^f
K_{max} = set of maximum cardinality cutsets
$K_i^f(\overline{K}_i^f)$ = fundamental cutset of spanning tree $T(\overline{T})$ containing branch b_i
n = number of nodes in the process graph
n_s = minimum no. of sensors required to observe all variables
p_i = failure probability of the sensor of variable i
p_m = sensor having the highest probability of failure
$R(i), \overline{R}(i)$ = reliability of variable i in spanning tree $T, \overline{T}$
S_i = sensor i
T = spanning tree of the process graph
$\overline{T}$ = updated spanning tree of T
$V(V')$ = set of nodes of graph $G(G')$

Other symbols

$\in$ = member of
$|\cdot|$ = cardinality of
$\oplus$ = ring sum of
Π = product of
$\equiv$ = identically equal
$\rightarrow$ = implies that

Literature Cited

Aho, A. V., J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, New York (1974).
Crowe, C. M., "Observability and Redundancy of Process Data for Steady State Reconciliation," *Chem. Eng. Sci.*, **44**, 2909 (1989).
Deo, N., *Graph Theory with Applications to Engineering and Computer Science*, Prentice Hall, Englewood Cliffs, NJ (1974).
Kretsovalis, A., and R. S. H. Mah, "Effect of Redundancy on Estimation Accuracy in Process Data Reconciliation," *Chem. Eng. Sci.*, **42**, 2115 (1987a).
Kretsovalis, A., and R. S. H. Mah, "Observability and Redundancy Classification in Multicomponent Process Network," *AIChE J.*, **33**, 910 (1987b).
Kretsovalis, A., and R. S. H. Mah, "Observability and Redundancy Classification in Generalized Process Networks: I. Theorems," *Comput. Chem. Eng.*, **12**, 671 (1988).
Mah, R. S. H., G. M. Stanley, and D. M. Downing, "Reconciliation and Rectification of Process Flow and Inventory Data," *Ind. Engng. Chem. Process Des. Dev.*, **15**, 175 (1976).
Nijenhuis, A., and H. S. Wilf, *Combinatorial Algorithms for Computers and Calculators*, Academic Press, New York (1978).
Omatu, S., and J. H. Seinfeld, *Distributed Parameter Systems: Theory and Applications*, Oxford University Press, Oxford (1989).
Serth, R. W., and W. A. Heenan, "Gross Error Detection and Data Reconciliation in Steam-Metering Systems," *AIChE J.*, **32**, 733 (1986).
Stanley, G. M., and R. S. H. Mah, "Observability and Redundancy in Process Data Estimation," *Chem. Eng. Sci.*, **36**, 259 (1981a).
Stanley, G. M., and R. S. H. Mah, "Observability and Redundancy Classification in Process Networks, Theorems and Algorithms," *Chem. Eng. Sci.*, **36**, 1941 (1981b).
Vaclavek, V., and M. Loucka, "Selection of Measurements Necessary to Achieve Multicomponent Mass Balances in Chemical Plant," *Chem. Eng. Sci.*, **31**, 1199 (1976).

Appendix A: Graph-Theoretic Terminology

Graph and subgraph

An undirected (respectively directed) *graph* $G(V, E)$ consists of a set of objects $V = \{v_1, v_2, \ldots, v_n\}$ called vertices or nodes and another set $E = \{e_1, e_2, \ldots, e_m\}$ called edges, such that each edge e_k is identified with an unordered (respectively ordered) pair (v_i, v_j) which are called the end nodes. The edges are said to be incident on these nodes. Schematically, nodes are represented as points, and edges are represented by arcs joining these points as shown in Figure A1. Note that by its very definition, a graph must contain both the end nodes of every edge it contains.

A graph $G'(V', E')$ is said to be a *subgraph* of $G(V, E)$ if $V' \in V$ and $E' \in E$, and each edge of G' has the same end vertices in G' as in G. For example, the graph shown in Figure A2 is a subgraph of $G(V, E)$ shown in Figure A1.

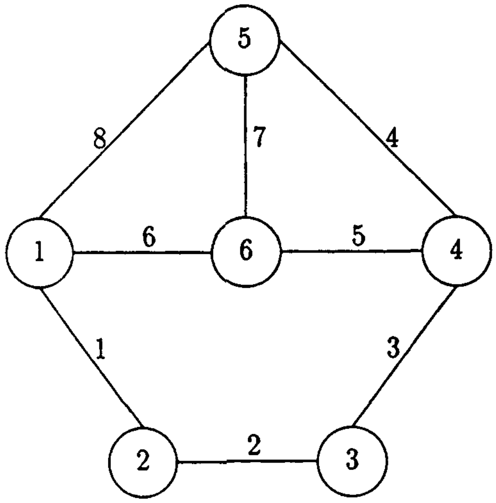

Figure A1. An undirected graph *G*.

Paths, cycles, and connectivity

A *path* between vertices v_o and v_l is an alternating sequence of distinct vertices and edges $v_o\, e_o\, v_1\, e_1\, v_2 \ldots\, v_{l-1}\, e_{l-1}\, v_l$ where (v_i, v_{i+1} are the end nodes of edge e_i). If $v_o = v_l$, then the path is called a *cycle*. For example, in Figure A1, the sequence of edges 1, 2 and 3 together with their end nodes is a path and the sequence 1, 2, 3, 5 and 6 together with their end nodes is a cycle.

A graph G is said to be *connected* if there is at least one path between every pair of vertices in G. The graph in Figure A1 is connected.

Trees, spanning trees, branches, and chords

A *tree* is a connected graph that does not contain any cycle. The graph shown in Figure A3 is a tree. A tree T, is said to be a *spanning tree* of graph G, if it is a subgraph of G and all vertices of G are also contained in T. For example, the graph shown in Figure A4 is a spanning tree of the graph in Figure A1, whereas that shown in Figure A3 is not. An edge in a spanning tree T is called a *branch* of T, while an edge of G which is not in T is called a ***chord***. Note that branches and chords are defined with respect to a spanning tree. For example, edges 1, 3, 4, 7, and 8 shown in the spanning tree of Figure A4 are branches while edges 2, 5 and 6 which are present in Figure A1 but not in spanning tree of Figure A4 are chords.

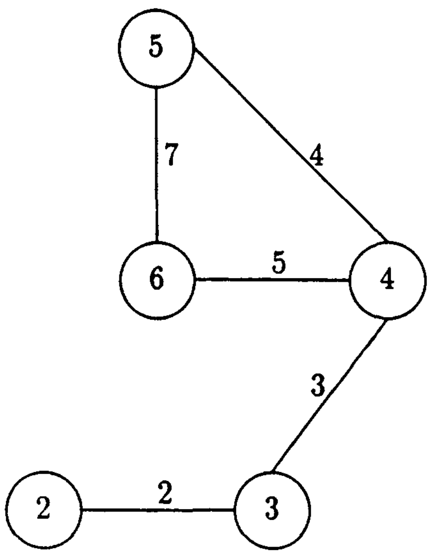

Figure A2. A subgraph of *G*.

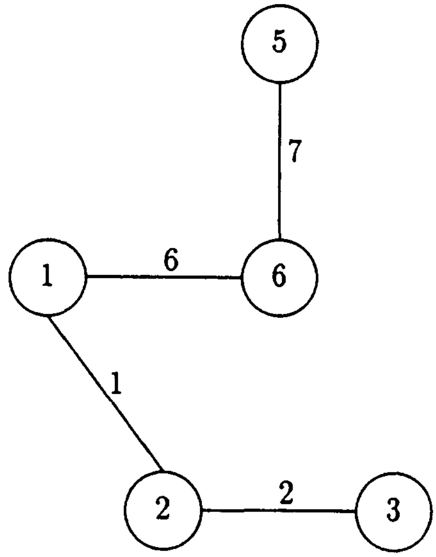

Figure A3. A tree.

Cutsets, fundamental cutsets, and ring sum

A *cutset* of a connected graph G, is a set of edges whose removal from G disconnects it, but the removal of a proper subset of these edges does not disconnect G. For example in Figure A1, the set of edges 3, 6, 8 is a cutset. However, edges 2, 3, 6, 8 does not form a cutset (although, their removal disconnects G) since the removal of a proper subset of edges 3, 6, and 8 itself can disconnect G.

Fundamental cutsets are defined with respect to a spanning tree T of G. A fundamental cutset is a cutset of G which contains exactly one branch of T. For example, in Figure A1,

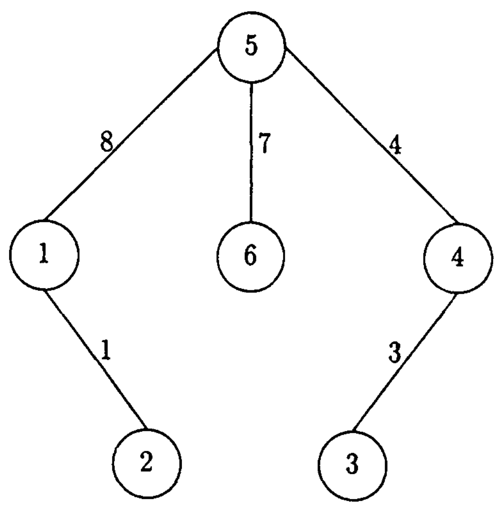

Figure A4. A spanning tree of *G*.

edges 2, 6 and 8 form a fundamental cutset with respect to the spanning tree in Figure A4 where edge 8 is a branch and all the remaining edges are chords. On the other hand the set of edges 4, 5, 6 and 8 is not a fundamental cutset with respect to the spanning tree shown in Figure A4 (although, it is a cutset) since it contains more than one branch of T.

The *ring sum* (denoted as $\oplus$) of two cutsets K_1 and K_2 is the set of all edges which are either in K_1 and K_2 but not in both. For example, the ring sum of two cutsets (1, 4, 5) and (4, 5, 6, 8) is a set (1, 6, 8). It does not contain edges 4 and 5 as these are common to both cutsets.

Appendix B: Proof of Lemmas

Proof of lemma 1

The proof of this lemma is given in Deo (1974).

Proof of lemma 2

Consider a spanning tree T of a graph G with branches $\{b_1, \cdots, b_{n-1}\}$ and chords $\{c_1, \cdots, c_{e-n+1}\}$. Without loss of generality, let fundamental cutsets K_x^f and K_y^f be defined as:

$$K_x^f = \{b_x, c_1, \cdots, c_r, c_{r+1}, \cdots, c_s\}$$
$$K_y^f = \{b_y, c_1, \cdots, c_r, c_{s+1}, \cdots, c_t\}$$

where chords $\{c_1, \cdots, c_r\}$ are common to both K_x^f and K_y^f.

The ring sum of K_x^f and K_y^f is the set of all edges in K_x^f and K_y^f excluding the common ones. Thus,

$$K_x^f \oplus K_y^f = \{b_x, c_{r+1}, \cdots, c_s, b_y, c_{s+1}, \cdots, c_t\}$$

All that is required to be proved is that if a proper subset of edges from the above set is deleted, it does not disconnect G. Deletion of $\{c_{r+1}, \cdots, c_s, c_{s+1}, \cdots, c_t\}$ will not disconnect G, because all these are chords and the branches of T still exist which maintains connectivity of G. Similarly deletion of b_x or b_y and $\{c_{r+1}, \cdots, c_s, c_{s+1}, \cdots, c_t\}$ will not disconnect G because common edges $\{c_1, \cdots, c_r\}$ preserve the connectivity of G. It is only when b_x and b_y and all the chords are deleted that G is disconnected. Hence the ring sum forms another cutset and not a union of edge disjoint cutsets.

Proof of lemma 3

Consider the measurement with highest sensor failure probability p_m [thus, the least reliability among measured variables is $(1-p_m)$]. This will be a chord of the spanning tree corresponding to the sensor network design. Since every chord appears in some fundamental cutset (Deo, 1974), let chord p_m be a member of fundamental cutset K_i^f which includes branch b_i and one or more additional chords. Thus,

$$R(b_i) = \prod_{\substack{j \in K_i^f \\ j \neq 1}} (1-p_j) \leq (1-p_m)$$

Either b_i has the lowest reliability or some other branch. In any case, the least reliability is attained for an unmeasured variable.

Proof of lemma 4

By lemma 1, we know that by placing a sensor on b_q and removing the sensor from chord $c_p \in K_q^f$, another spanning tree solution is obtained. Let the new spanning tree be $\overline{T}$ and let $\overline{R}(\cdot)$ represent the reliabilities of the variables and $\overline{K}_i^f$ be the fundamental cutsets corresponding to $\overline{T}$. Our objective is to prove that reliabilities of all variables in the new solution $\overline{T}$ are greater than $R(b_x)$. Let

$$K_x^f = \{b_x, c_1, \cdots, c_r, c_{r+1}, \cdots, c_s\}$$
$$K_q^f = \{b_q, c_1, \cdots, c_r, c_{s+1}, \cdots, c_t\}$$

Then

$$K = K_x^f \oplus K_q^f = \{b_x, c_{r+1}, \cdots, c_s, b_q, c_{s+1}, \cdots, c_t\}$$

We know from lemma 2 that K is a cutset of the graph. Furthermore, in the new spanning tree solution, cutset K contains only one unmeasured variable (b_x). It should be noted that condition (1) of this lemma ensures that the new unmeasured variable (c_p) is not a member of K. Based on these observations we conclude that K is a fundamental cutset with respect to spanning tree $\overline{T}$ containing branch b_x. Thus,

$$\overline{K}_x^f \equiv K$$

From condition 2 and the assumption that all sensors have same failure probability it follows that:

$$\overline{R}(b_x) > R(b_x)$$

For the new unmeasured variable c_p we get:

$$\overline{K}_p^f \equiv K_q^f \Rightarrow \overline{R}(c_p) = R(b_q) > R(b_x)$$

If c_p is a member of any fundamental cutset K_j^f then,

$$\overline{K}_j^f = K_j^f \oplus K_q^f$$

From condition 3, it therefore follows that:

$$\overline{R}(b_j) > R(b_x)$$

On the other hand, if c_p is not a member of cutset K_j^f, then the reliability of b_j remains unchanged (since $\overline{K}_j^f \equiv K_j^f$). Thus, the reliability of all variables corresponding to $\overline{T}$ is strictly greater than the minimum reliability corresponding to T. Thus,

$$\min_i \overline{R}(b_i) > R(b_x)$$

proving that the network reliability has improved.

Manuscript received Jan. 23, 1992, and revision received Aug. 7, 1992.