Efficient Procedure for the Design and Upgrade of Sensor Networks Using Cutsets and Rigorous Decomposition

Mayur Gala and Miguel J. Bagajewicz*

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

We present a decomposition procedure that accelerates the computational performance of the tree enumeration method presented by Gala and Bagajewicz [Gala, M.; Bagajewicz, M. *Ind. Eng. Chem. Res.* **2006**, 6679−6686], which is based on the union of cutsets. The new algorithm, which is rigorous, uses union and a ring sum of cutsets and is based on the decomposition of the graph into subgraphs. The robustness and the efficiency of the proposed algorithm are tested on a large scale refinery problem.

Introduction

In our first part,[1] we developed a tree algorithm to design a robust sensor network handling precision, residual precision, error detection, and resilience. The proposed tree algorithm included finding all the cutsets of the process graph and using these cutsets to explore solutions using tree enumeration, branching, and bounding. The proposed algorithm solved medium size problems efficiently and much faster than the original algorithm proposed by Bagajewicz,[2,3] which is based on streams. However, in the case of large size problems, the proposed algorithm proved to be inefficient due to the presence of a large number of cutsets.

To test the union of cutsets algorithm on a large size problem, a refinery problem (Figure 8) was attempted using the proposed algorithm based on a tree of cutsets. A total of 803 cutsets were found. The proposed algorithm explored a total of 153 857 760 nodes in 7 days and skipped 7.3×10^{240} nodes when it was interrupted. Assuming the same fraction of nodes to be skipped from the remaining nodes, the estimated time of completion is 45 days. In the present article, we propose a new concept of decomposition of the process flow diagram, reducing the total number of cutsets of the system used in the cutset algorithm.

The article is organized as follows: We first introduce the decomposition procedure. Then, we show that the decomposition method reduces the total number of cutsets and by using these reduced cutsets one can reconstruct each cutset of the entire undecomposed process flow diagram. The reduction in the number of cutsets increases with the number of decompositions, and it also changes with the locations of the chosen cuts for the decomposition. However, with the increase in the number of decompositions, the number of streams connecting the subdiagrams also increases, reducing the lower bound used in the stopping criteria of the new tree algorithm. We finally propose an approach to find the optimal location and number of decompositions in the flow diagram. We also discuss how the branching and stopping criteria used in the cutset algorithm are modified for the decomposition algorithm. The efficiency and robustness of the proposed algorithm are examined on a large scale refinery problem.

Decomposition of Process Flow Diagrams. Consider the simple process flow diagram shown in Figure 1. Table 1 lists all the cutsets of the flow diagram (Figure 1).

Let us decompose the process flow diagram of Figure 1 at the center (stream S_6) into two subflow diagrams (shown in Figure 2). We call the stream (or streams) that connect the subgraphs "connecting streams" (S_6 in our example). Table 2 lists the cutsets for each of decomposed subflow diagrams A and B.

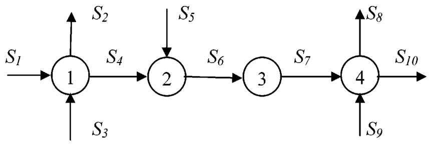

Figure 1. Ten stream process flow diagram.

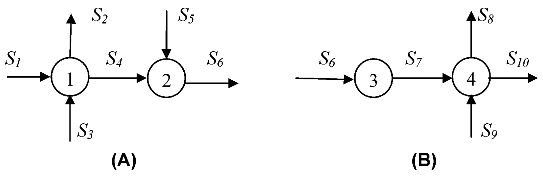

Figure 2. Single decomposition of Figure 1.

Table 1. List of All Cutsets for Figure 1

cutsets	streams
C_1	S_1, S_2, S_3, S_4
C_2	S_1, S_2, S_3, S_5, S_6
C_3	S_1, S_2, S_3, S_5, S_7
C_4	$S_1, S_2, S_3, S_5, S_8, S_9, S_{10}$
C_5	S_4, S_5, S_6
C_6	S_4, S_5, S_7
C_7	$S_4, S_5, S_8, S_9, S_{10}$
C_8	S_6, S_7
C_9	S_6, S_8, S_9, S_{10}
C_{10}	S_7, S_8, S_9, S_{10}

Table 2. List of Cutsets for the Subdiagrams of Figure 2

decomposed subflow diagram A		decomposed subflow diagram B	
cutsets	streams	cutsets	streams
C_{A1}	S_1, S_2, S_3, S_4	C_{B1}	S_6, S_7
C_{A2}	S_1, S_2, S_3, S_5, S_6	C_{B2}	S_6, S_8, S_9, S_{10}
C_{A3}	S_4, S_5, S_6	C_{B3}	S_7, S_8, S_9, S_{10}

The total number of the cutsets of the system consisting of two decomposed subflow diagram reduces from 10 to 6.

A very well-known property of graph theory is that the ring sum (union − intersection) of any two cutsets of a graph, which has at least one stream in common, results in a new cutset of the same graph. Hence, the ring sum on any two cutsets, each from a different subdiagram and both having the connecting streams in common, results in a new cutset containing streams of both decomposed subflow diagrams. We illustrate this next.

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

10.1021/ie050866j CCC: $33.50 © 2006 American Chemical Society
Published on Web 08/25/2006

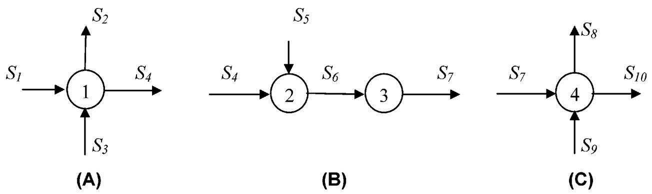

Figure 3. Two decompositions of Figure 1.

Table 3. List of Cutsets of Subdiagrams of Figure 3

Decomposed Subflow Diagrams					
A		B		C	
cutsets	streams	cutsets	streams	cutsets	streams
C_{A1}	S_1, S_2, S_3, S_4	C_{B1}	S_4, S_5, S_6	C_{C1}	S_7, S_8, S_9, S_{10}
		C_{B2}	S_4, S_5, S_7		
		C_{B3}	S_6, S_7		

The cutsets of Table 1 that are not listed in Table 2 are C_3 $\{S_1, S_2, S_3, S_5, S_7\}$, C_4 $\{S_1, S_2, S_3, S_5, S_8, S_9, S_{10}\}$, C_6 $\{S_4, S_5, S_7\}$, and C_7 $\{S_4, S_5, S_8, S_9, S_{10}\}$. These cutsets contain streams present in both subflow diagrams. We now show the ring sum operations to obtain these cutsets:

$$C_3\ \{S_1, S_2, S_3, S_5, S_7\} = C_{A2}\ \{S_1, S_2, S_3, S_5, S_6\} \times C_{B1}\ \{S_6, S_7\}$$

$$C_4\ \{S_1, S_2, S_3, S_5, S_8, S_9, S_{10}\} = C_{A2}\ \{S_1, S_2, S_3, S_5, S_6\} \times C_{B2}\ \{S_6, S_8, S_9, S_{10}\}$$

$$C_6\ \{S_4, S_5, S_7\} = C_{A3}\ \{S_4, S_5, S_6\} \times C_{B1}\ \{S_6, S_7\}$$

$$C_7\ \{S_4, S_5, S_8, S_9, S_{10}\} = C_{A3}\ \{S_4, S_5, S_6\} \times C_{B2}\ \{S_6, S_8, S_9, S_{10}\}$$

Now, let us decompose the process flow diagram of Figure 1 into three subdiagrams cutting at streams S_4 and S_7. Streams S_4 and S_7 are now the connecting streams between the decomposed subdiagrams (Figure 3). Table 3 lists all the cutsets of the decomposed subflow diagrams.

The total number of cutsets of the system reduces to five. The cutsets from Table 1 not listed in Table 3 are C_2 $\{S_1, S_2, S_3, S_5, S_6\}$, C_3 $\{S_1, S_2, S_3, S_5, S_7\}$, C_4 $\{S_1, S_2, S_3, S_5, S_8, S_9, S_{10}\}$, C_7 $\{S_4, S_5, S_8, S_9, S_{10}\}$, and C_9 $\{S_6, S_8, S_9, S_{10}\}$. Again, these cutsets can be obtained by performing a ring sum operation on the decomposed cutsets of Table 3.

$$C_2\ \{S_1, S_2, S_3, S_5, S_6\} = C_{A1}\ \{S_1, S_2, S_3, S_4\} \times C_{B1}\ \{S_4, S_5, S_6\}$$

$$C_3\ \{S_1, S_2, S_3, S_5, S_7\} = C_{A1}\ \{S_1, S_2, S_3, S_4\} \times C_{B2}\ \{S_4, S_5, S_7\}$$

$$C_4\ \{S_1, S_2, S_3, S_5, S_8, S_9, S_{10}\} = C_{A1}\ \{S_1, S_2, S_3, S_4\} \times C_{B2}\ \{S_4, S_5, S_7\} \times C_{C1}\ \{S_7, S_8, S_9, S_{10}\}$$

$$C_7\ \{S_4, S_5, S_8, S_9, S_{10}\} = C_{B2}\ \{S_4, S_5, S_7\} \times C_{C1}\ \{S_7, S_8, S_9, S_{10}\}$$

$$C_9\ \{S_6, S_8, S_9, S_{10}\} = C_{B3}\ \{S_6, S_7\} \times C_{C1}\ \{S_7, S_8, S_9, S_{10}\}$$

Finally, if we decompose the flow diagram of Figure 1 into the maximum number of subdiagrams possible (4; see Figure 4). The total number of cutsets of the system is now four. Table 4 lists the cutsets of the subdiagrams. As in previous illustrations, all the cutsets of Table 1 not listed in Table 4 can be explored by performing the ring sum operation on the cutsets of Table 4.

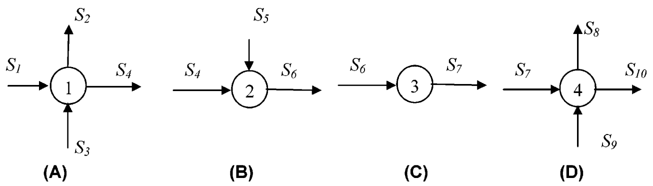

Figure 4. Three decompositions of Figure 1.

Table 4. List of Cutsets of Subdiagrams of Figure 4

cutsets	streams
C_{A1}	S_1, S_2, S_3, S_4
C_{B2}	S_4, S_5, S_6
C_{C3}	S_6, S_7
C_{D4}	S_7, S_8, S_9, S_{10}

Table 5. Total Number of Cutsets of the System with Single Decomposition at Different Locations

connecting streams	total number of cutsets of the system
S_4	7
S_6	6
S_7	7

With a fixed number of decompositions, the total number of cutsets of the system also varies with the location of decomposition, as illustrated in Table 5 for Figure 1, where the total number of cutsets of the system for decomposition at different locations (different connecting streams) is depicted.

On the basis of the above, we now propose a tree searching algorithm similar to the one developed by Bagajewicz[1] and used by Gala and Bagajewicz,[1] but now, we use the reduced number of cutsets obtained from decomposition of the flow diagram as building blocks. We also had to revise the branching and stopping criteria.

Tree Enumeration Algorithm with Decomposition

In our previous article, we proposed a tree enumerating method with branching and bounding similar to the one developed by Bagajewicz[2] for streams. The method consisted of using all the cutsets of the process graph to develop the tree where each node was defined by a union of cutsets rather than the union of stream measurements. In the present article, we propose a similar tree enumerating method but now using the reduced numbers of cutsets obtained from decomposition of the process flow diagram and different operations than the simple union at each node.

The branching and stopping criteria used were also modified as follows: While considering the design of the redundant sensor network in a tree, we measure all the variables of the cutsets active in that node, and for a nonredundant network (considering observability), we measure all the variables of the active cutsets except the key variables. Next, we explain the operations performed at each node of the enumeration tree when exploring the cutsets obtained from decomposition.

Single Node Operation in the Enumeration Tree. Using the reduced number of cutsets obtained from the decomposition of the process flow diagram, we construct all the remaining cutsets of the system that can be constructed by performing the ring sum operation of the cutsets selected in that node. In other words, various solutions can emerge from considering this node: the union of all cutsets, which is what is done in the previous algorithm[1] and in the different combinations of ring sum operations and unions (the ring sums add all the remaining cutsets that contain elements of the two subgraphs involved). We now illustrate this.

Consider the flow sheet of Figure 1, which is decomposed into the two subdiagrams (with connecting stream S_6) shown in Figure 2. The list of candidate cutsets to use is the one of Table 2. Assume a node in a cutset tree containing cutsets C_{A2} $\{S_1, S_2, S_3, S_5, S_6\}$ and C_{B1} $\{S_6, S_7\}$. We first perform the union operation to get $\{S_1, S_2, S_3, S_5, S_6, S_7\}$ and evaluate the node with $q = \{1110111000\}$. Next, if the node contains cutsets from two different subdiagrams intersecting at the connecting stream, as is the case, then we perform the ring sum operation of the cutsets to obtain one of the remaining cutsets, $\{S_1, S_2, S_3, S_5, S_7\}$ and evaluate the node using $q = \{1110101\}$. Note that if the cutsets do not intersect at the connecting stream, then the ring sum is equal to the union, which is the first operation tried. It is important to point out that we only perform ring sum operations on cutsets that belong to different subgraphs and perform only a union when the cutsets belong to the same subgraph. For example, in the case of Figure 2, consider that in a node we have cutsets C_{A2} $\{S_1, S_2, S_3, S_5, S_6\}$, C_{B1} $\{S_6, S_7\}$, and C_{B2} $\{S_6, S_8, S_9, S_{10}\}$. We first perform the union operation to get $\{S_1, S_2, S_3, S_5, S_6, S_7, S_8, S_9, S_{10}\}$ and evaluate the node with $q = \{1110111111\}$. Then, we perform the following ring sum $C_{A2} \times \{C_{B1} \cup C_{B2}\}$ to obtain $\{S_1, S_2, S_3, S_5, S_7, S_8, S_9, S_{10}\}$ with the corresponding $q = \{1110101111\}$.

If we have more than two decompositions and the node contains connecting streams and cutsets from different subdiagrams, then we perform all combinations of union and ring sum operations between them. Consider again the flow sheet of Figure 1 but now decomposed into three subdiagrams (with connecting streams S_4 and S_7), as shown in Figure 3 (cutsets listed in Table 3). Assume a node in a tree containing the cutsets C_{A1} $\{S_1, S_2, S_3, S_4\}$, C_{B1} $\{S_4, S_5, S_6\}$, and C_{C1} $\{S_7, S_8, S_9, S_{10}\}$. The node contains cutsets from each subflow diagram and all the connecting streams. The combination of union and ring sum operations performed between the cutsets is as follows:

$$C_{A1} \cup C_{B1} \cup C_{C1} = \{S_1, S_2, S_3, S_4, S_5, S_6, S_7, S_8, S_9, S_{10}\} \quad (q = 1111111111)$$

$$C_{A1} \times C_{B1} \cup C_{C1} = \{S_1, S_2, S_3, S_5, S_6, S_7, S_8, S_9, S_{10}\} \quad (q = 1110111111)$$

$$C_{A1} \cup C_{B1} \times C_{C1} = \{S_1, S_2, S_3, S_4, S_5, S_6, S_8, S_9, S_{10}\} \quad (q = 1111110111)$$

$$C_{A1} \times C_{B1} \times C_{C1} = \{S_1, S_2, S_3, S_5, S_6, S_8, S_9, S_{10}\} \quad (q = 1110110111)$$

The node is evaluated for every vector q, and the feasible one with the minimum cost is considered as the output solution of the node.

The tree enumeration procedure (shown in Figure 5) is similar to the one we proposed in our previous article.

The proposed procedure is as follows:

(1) Decompose the process flow diagram at fixed locations into some fixed subflow diagrams. Find the cutsets of each subflow diagram and add them to get a reduced number of cutsets of the system. If nonredundant networks are allowed, add the measurement of individual key variables as one candidate to use and remove the key variables from all the cutsets that have been identified. When a cutest contains more than one key variable, then create as many cutsets measuring all but each key variable at the time. The list now has cutsets with key variables removed and key variables.

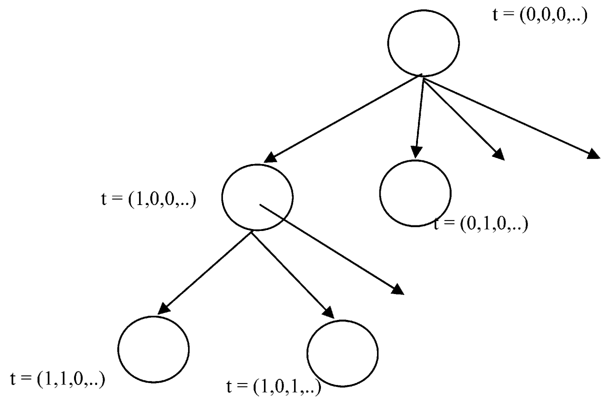

Figure 5. Enumeration tree using cutsets.

(2) Sort all building block candidates (individual measurements and cutsets) in ascending order of their cost with a maximum number of key variables first. The idea is that one would like to capture all the cutsets that contain the key variables first and then make the union of these with others. If we add cutsets without key variables, we believe that this will result in an inefficient search. (The cost of a cutset is equal to sum of the costs of the instruments placed on the streams of that cutset).

(3) Start with the root node with no cutsets (candidates) 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 (candidate) at a time which is chosen from the remaining cutsets (candidates) using a branching criterion.

(5) While implementing 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 and ring sum operation of different sets 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. Using the ring sum operation at each node brings the possibility of reducing the cost of the node considered. Our intention here is to choose the cutset to add that will contribute to the smallest overall cost. In our previous article, we performed only union operations at each node resulting in the cost of the nodes always increasing when adding cutsets Because we perform both union and ring sum operations at each node, the branching criterion needs modification. We therefore use the following.

While exploring the tree from one node to the other, either going down the tree or exploring the sister node: (i) the union operation between the existing cutsets and all new candidate cutsets is performed; (ii) all the connecting streams present are eliminated; (iii) the cost is obtained; (iv) the candidate that results in the lowest cost is selected.

Stopping Criterion. We discussed in previous section that at each node we perform combinations of union and ring sum operations, evaluate the node with all q vectors, and select the one which is feasible with the lowest cost. If the selected q vector does not contain any connecting streams, then all the nodes down the tree or in sister branches will have higher cost. This is because in exploring nodes down the tree or sister nodes, at least one stream will be added to the existing feasible set of streams and therefore the cost will always increase. Hence, if the selected vector q is feasible and does not contain any

Table 6. Flow Rates for Example 1

streams	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}
flow	150.2	86	82	146.2	52	198.2	198.2	85	47	160.2

Table 7. Costs of Flow Meters for Example 1

streams	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}
cost	2300	2500	2200	1400	1400	2300	2900	2500	2400	2200

Table 8. Sorted Cutsets for Example 1

cutset	streams	no. of key variables	cost
C_{B1}	S_4, S_5, S_6	1	5 100
C_{B3}	S_6, S_7	1	5 200
C_{C1}	S_7, S_8, S_9, S_{10}	1	10 000
C_{B2}	S_4, S_5, S_7	0	5 700
C_{A1}	S_1, S_2, S_3, S_4	0	8 400

connecting stream, the tree is not explored further down on any branches but backs up two levels and explores the rest of the tree.

However, if the selected vector q contains connecting streams, then the nodes down the tree or in sister branches can have vectors q with lower cost. This may be due to the addition of cutsets (with or without connecting streams) and then performing ring sum operations, eliminating all connecting streams to give lower cost. Hence, we use the following lower bound stopping condition:

$$\text{Current feasible node cost} - \sum \text{Connecting streams cost in this node} + \text{Min instrument cost} \geq \text{Best feasible node cost found}$$

where the "minimum instrument cost" is the minimum value of all costs of all candidate instruments in the network. This condition basically calculates a lower bound to which the cost can be reduced in some node down the tree or in sister nodes. The condition emerges from the extreme case in which new cutsets are added such that all of them contain the connecting streams and all existing measured streams except one, which could be the stream with the lowest cost. Why is this a lower bound? First, the cost can only go down because of ring sum operations, which only eliminate connecting streams (recall that we do not perform ring sum operations on cutsets belonging to the same subgraph). A lower bound obtained by eliminating all the connecting stream costs and not adding any new stream does not exist, because at least one cutset has to be different than the existing ones. Thus, the lowest cost that could be added is the one corresponding to a cutset that contains the stream with the minimum cost and the rest of the streams already existing in the current node.

We illustrate the performance of the procedure with several examples.

Examples

Example 1. Consider the 10 stream process flow diagram with 4 units shown in Figure 1. We use this example just to illustrate the cutset tree procedure, not the method of finding the optimal locations and number of decompositions or the computational performance. The flow rates of Table 6 were used. It is assumed that flow meters of 1.5% precision are available. Table 7 gives the costs.

Precision is only required for variables S_6 and S_8 with $\sigma_6{}^* = 2\%$ and $\sigma_8{}^* = 2\%$. A residual precision order of 1 ($k = 1$) and threshold of $\psi_6{}^* = 4\%$ and $\psi_8{}^* = 4\%$ are required. We consider the two decompositions of example 1 shown in Figure 3 at streams S_4 and S_7. Next, we find the cutsets of all three subflow

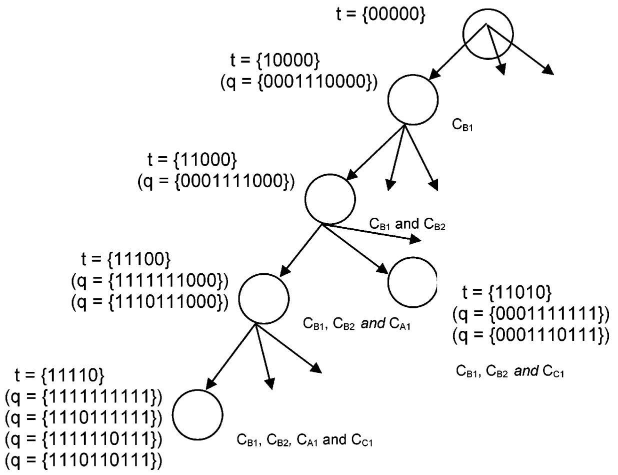

Figure 6. Tree for example 1.

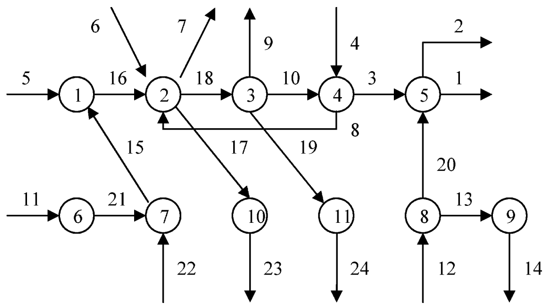

Figure 7. Example 2 (following the work of Madron and Veverka[4]).

diagrams and add them to get the cutsets of the system. A total of five cutsets are found; these cutsets are sorted in ascending order of cost with those containing more key variables first and so on. First, three cutsets containing a single key variable are sorted in ascending order of their cost, and then, the remaining two cutsets with no key variables are sorted. These sorted cutsets are listed in Table 8, which we use to explore the tree.

We start the tree with a root node with no cutsets added (trivially infeasible). Now, we start exploring the tree by adding the first cutset and we pick the cheapest on C_{B1} (Figure 6); thus, node $t = \{10000\}$ is evaluated by adding the instruments on the streams S_4, S_5, and S_6 ($q = \{0001110000\}$). The node is infeasible, so we go to the next level in the tree, $t = \{11000\}$. Here, cutset C_{B1} is active and we select a new cutset to add from the remaining four cutsets using the branching criteria. The final cost after adding the cutsets are $C_{B1} \cup C_{B3} = 8000$, $C_{B1} \times C_{C1} = 12\,200$, $C_{B1} \cup C_{B2} = 8000$, and $C_{B1} \times C_{A1} = 8500$. $C_{B1} \cup C_{B3}$ and $C_{B1} \cup C_{B2}$ give the same resultant cost. Thus, cutset C_{B2} is selected and added to cutset C_{B1}, and instruments are placed on streams S_4, S_5, S_6, S_7 ($q = \{0001111000\}$). The node is infeasible; hence, we go deeper into the tree, selecting one more cutset to be added from the remaining three cutsets.

At node $t = \{11100\}$, C_{B1} and C_{B2} are the active cutsets and we need to select and add one cutset from the remaining three cutsets, C_{B3}, C_{A1}, and C_{C1}. The one that provides the lowest cost after elimination of the connecting stream is C_{A1}; indeed, $C_{B1} \cup C_{B2} \cup C_{B3}$ gives rise to $q = \{0001111000\}$, that is, streams S_4, S_5, S_6, and S_7, which were already considered in the previous node; hence, we discard it. Now, the cutset C_{A1} is from a different subflow diagram; hence, we perform a ring sum and a union operation. We evaluate the node with $q = \{1110111000\}$ ($C_{B1} \cup C_{B2} \times C_{A1}$) (ring sum operation; stream S_4 is eliminated). The node is also infeasible if the union is

Table 9. Flow Rates for 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 10. Cost of Flow Meters for Example 2

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

chosen. The node is again infeasible; hence, we go still deeper into the tree. The next choices are C_{B3} and C_{C1}, in which again $C_{B1} \cup C_{B2} \times C_{A1} \cup C_{B3}$ have been already considered in a previous node; hence, we choose C_{C1}. Here, we have cutsets from all three subflow diagrams; hence, we perform all combinations of union and ring sum operations evaluating the node with $q = \{1111111111\}$ (union operation), $q = \{1110111111\}$ (ring sum operation eliminating stream S_4), $q = \{1111110111\}$ (ring sum operation eliminating stream S_7), and $q = \{1110110111\}$ (ring sum operation eliminating both streams S_4 and S_7). The node is feasible for all four combinations; hence, we choose $q = \{1110110111\}$ with a minimum cost of 17 800. Now, no cutsets are left to add or even explore the sister nodes; hence. we back up two levels and explore the sister nodes.

Table 11. Cutsets for Single Decomposition of Example 2

connecting streams	total cutsets
$\{S_{21}\}$	92
$\{S_{15}\}$	70
$\{S_{16}\}$	50
$\{S_{18}, S_8\}$	36
$\{S_{10}, S_8\}$	44
$\{S_3\}$	59
$\{S_{20}\}$	76
$\{S_{13}\}$	95

At node $t = \{11010\}$, we have C_{B1} and C_{B2} active. At this point, C_{A1} has already been considered. Cutsets C_{B3} and C_{C1} are the available cutsets, and we choose and add cutset C_{C1} which gives a lower cost of 12 200 after elimination of the connecting streams. Indeed, C_{C1} is from a different subflow diagram; hence, we evaluate the node with $q = (0001111111)$ and $q = (0001110111)$. The node is feasible with $q = (0001110111)$, measuring streams S_4, S_5, S_6, S_8, S_9, and S_{10} with a cost of 12 200. This is the optimal node found, but we continue with the rest of the tree with the same branching and stopping criteria (see Figure 6). The algorithm explores a total of 17 nodes out of 311 nodes (the total number of nodes $= 2^{no\ of\ \text{cutsets}} - 1$ or $2^5 - 1$). The algorithm took less than 1 s to solve this problem on an Intel Celeron 1.39 GHz, 256 MB RAM PC.

Optimal Number and Locations of Decompositions. We showed in the previous section that with the increase in the

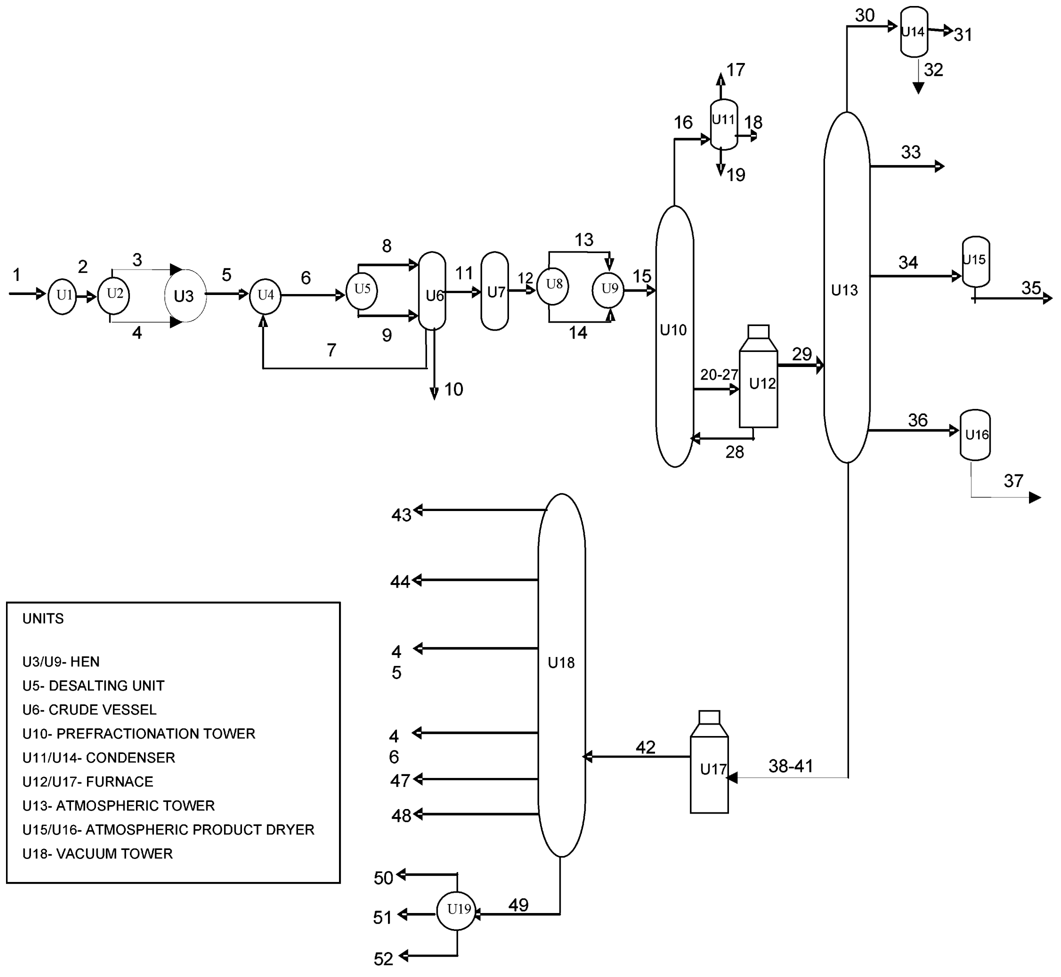

Figure 8. Crude distillation unit flow diagram. Reprinted with permission from ref 5. Copyright 2005 John Wiley & Sons.

Table 12. Results for Example 2 for Different Numbers of Decompositions

no. of subgraphs	total cutsets	connecting streams	connecting streams cost	time	nodes explored	optimal node
0	116	none		1 min 48 s	4 219	832
1	36	$\{S_{18}, S_8\}$	24	1 min 17 s	15 923	11 391
2	26	$\{S_{16}\},\{S_{10}, S_8\}$	38	43 s	10 462	236
3	21	$\{S_{15}\},\{S_{18}, S_8\},\{S_3\}$	54	1 min 55 s	14 787	7 396
3	21	$\{S_{16}\},\{S_{18}, S_8\},\{S_3\}$	56	59 s	11 178	249
4	18	$\{S_{15}\},\{S_{18}, S_8\},\{S_{10}, S_8\},\{S_{20}\}$	69	1 min 47 s	12 538	5 953
4	18	$\{S_{16}\},\{S_{18}, S_8\},\{S_{10}, S_8\},\{S_{20}\}$	71	1 min 24 s	10 133	91
5	16	$\{S_{15}\},\{S_{17}\},\{S_{18}, S_8\},\{S_{10}, S_8\},\{S_{20}\}$	86	2 min 25 s	10 898	4 590

Table 13. Mass Flow Rates for Example 3

streams	mass flow rate	streams	mass flow rate
S_1	413 349	S_{27}	60 413
S_2	419 579	S_{28}	103 939
S_3	209 316	S_{29}	386 580
S_4	210 262	S_{30}	57 169
S_5	419 579	S_{31}	45 829
S_6	460 520	S_{32}	4 202
S_7	26 510	S_{33}	26 133
S_8	230 650	S_{34}	73 900
S_9	229 870	S_{35}	73 704
S_{10}	26 243	S_{36}	50 851
S_{11}	413 650	S_{37}	50 715
S_{12}	413 650	S_{38}	45 902
S_{13}	206 932	S_{39}	45 878
S_{14}	206 717	S_{40}	45 928
S_{15}	413 650	S_{41}	45 851
S_{16}	27 068	S_{42}	185 593
S_{17}	5 124	S_{43}	38 557
S_{18}	21 467	S_{44}	18 932
S_{19}	478	S_{45}	19 846
S_{20}	61 562	S_{46}	23 880
S_{21}	60 985	S_{47}	18 196
S_{22}	61 253	S_{48}	18 106
S_{23}	61 490	S_{49}	48 081
S_{24}	61 109	S_{50}	15 154
S_{25}	60 796	S_{51}	20 268
S_{26}	62 012	S_{52}	12 659

Table 14. Costs of Flow Meters for Example 3

streams	cost	streams	cost
S_1	2000	S_{27}	2000
S_2	2000	S_{28}	1800
S_3	1800	S_{29}	1500
S_4	1800	S_{30}	2300
S_5	2200	S_{31}	2100
S_6	2100	S_{32}	1800
S_7	2100	S_{33}	2200
S_8	1700	S_{34}	2200
S_9	1700	S_{35}	2000
S_{10}	2400	S_{36}	2200
S_{11}	2000	S_{37}	2200
S_{12}	2000	S_{38}	2000
S_{13}	1800	S_{39}	2000
S_{14}	1800	S_{40}	2000
S_{15}	1500	S_{41}	2000
S_{16}	2300	S_{42}	2300
S_{17}	2200	S_{43}	1800
S_{18}	2200	S_{44}	1800
S_{19}	1800	S_{45}	1800
S_{20}	2000	S_{46}	2100
S_{21}	2000	S_{47}	2100
S_{22}	2000	S_{48}	2100
S_{23}	2000	S_{49}	2300
S_{24}	2000	S_{50}	2000
S_{25}	2000	S_{51}	2000
S_{26}	2000	S_{52}	2000

number of decompositions the total number of cutsets of the system decreases but the number of connecting streams (streams connecting the subflow diagram) increases. Also depending upon the topology of the flow diagram, for a certain fixed number of decompositions, the total number of cutsets and the number of connecting streams vary with the locations of decomposition. Important parameters which affect the decomposition algorithm are the total number of reduced cutsets and connecting stream costs (total number of connecting streams). By reducing the number of cutsets (by increasing the number of decompositions), the size of the tree gets smaller, which may solve the problem faster, but at the same time the number of connecting streams increases, which increases the cost of the connecting streams at nodes in the cutset tree. A larger value of the total connecting stream cost reduces the lower bound used in the stopping criteria and hence forces the algorithm to go deeper into the tree and explore more nodes. Because one may attempt to solve a particular problem more than once, we now propose a heuristic procedure to find the optimal number of decompositions and the locations for the same.

Table 15. Cutsets for Single Decomposition of Example 3

connecting streams	cutsets	connecting streams	cutsets
$\{S_2\}$	724	$\{S_{16}\}$	460
$\{S_3, S_4\}$	650	$\{S_{20}-S_{29}\}$	140
$\{S_5\}$	578	$\{S_{29}\}$	128
$\{S_6, S_7\}$	508	$\{S_{30}\}$	448
$\{S_7, S_8, S_9\}$	440	$\{S_{34}\}$	448
$\{S_{11}\}$	374	$\{S_{36}\}$	448
$\{S_{12}\}$	310	$\{S_{38}-S_{41}\}$	272
$\{S_{13}, S_{14}\}$	248	$\{S_{42}\}$	446
$\{S_{15}\}$	188	$\{S_{49}\}$	622

Procedure.

(1) Choose a maximum running time.

(2) Make a list of selected places for single and multiple decompositions. For example, for single decomposition, one would choose connecting streams such that the subgraphs have a similar number of units or cutsets.

(3) Perform a single decomposition on the given flow diagram at all the possible proposed locations and find the reduced number of cutsets for each. The problem is then solved at each location, and the computation time and the reduced number of cutsets are recorded. If the maximum time chosen is exceeded, the run is stopped.

(4) Increase the number of decompositions and repeat the procedure of solving the problem at the proposed locations.

(5) The trend expected is that the computation time always increases with an increase in the number of decompositions or the computation time first decreases until a certain number of decompositions is reached and then again increases. The lowest computation time obtained from a given number of decompositions becomes the lower bound time for performing further runs.

Example 2. The network shown in Figure 7 was proposed by Madron and Veverka.[4] We use this example to illustrate the procedure of finding the optimal locations and number of decompositions. It consists of 24 streams and 11 nodes. The flow rates of Table 9 and costs of Table 10 were considered. A precision of 2.5% is required for variables S_3, S_{10}, S_{16}, S_{17}, S_{20}, and S_{24} ($\sigma_i^* = 2.5\%$, $i = 3, 10, 16, 17, 20,$

Table 16. Results for Example 3 for Different Numbers of Decompositions

no. of subgraphs	total cutsets	connecting streams	connecting streams cost	time	nodes explored	optimal node
1	128	$\{S_{29}\}$	1500	93 min 45 s	415 836	2 666
2	86	$\{S_{12}\},\{S_{29}\}$	3500	5 min 4 s	12 990	856
3	68	$\{S_{11}\},\{S_{29}\},\{S_{42}\}$	5800	9 min 42 s	17 279	1 887
3	68	$\{S_{12}\},\{S_{29}\},\{S_{42}\}$	5800	8 min 21 s	10 653	108
4	56	$\{S_5\},\{S_{12}\},\{S_{29}\},\{S_{42}\}$	8000	12 min 34 s	19 472	739
4	56	$\{S_5\},\{S_{15}\},\{S_{29}\},\{S_{42}\}$	7500	12 min 34 s	18 304	1 312
5	47	$\{S_5\},\{S_{11}\},\{S_{15}\},\{S_{29}\},\{S_{42}\}$	9500	23 min 21 s	28 739	257

Table 17. Results for Example 3 for Two Decompositions at Different Locations

no. of subgraphs	total cutsets	connecting streams	connecting streams cost	time	nodes explored	optimal node
2	88	$\{S_{13},S_{14}\},\{S_{29}\}$	5100	6 min 4 s	7 277	536
2	92	$\{S_{15}\},\{S_{29}\}$	3000	4 min 12 s	3 700	291
2	112	$\{S_{12}\},\{S_{20}-S_{28}\}$	19 800	122 min 47 s	389 795	16 761
2	118	$\{S_{12}\},\{S_{38}-S_{41}\}$	10 000	42 min 22 s	87 635	2 013
2	110	$\{S_{29}\},\{S_{42}\}$	3 800	24 min 43 s	27 643	1 876
2	98	$\{S_5\},\{S_{29}\}$	3 700	16 min 33 s	18 635	1 342
2	118	$\{S_{29}\},\{S_{49}\}$	3 800	29 min 12 s	29 743	2 396

24). A residual precision 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.

We first decompose the flow diagram (Figure 7) into two subdiagrams at different locations along the length and find the location which gives the minimum number of cutsets. One method to automatically find the optimal locations for each number of decompositions is presented in the Appendix. The proposed method converts the incidence matrix into a diagonally dense matrix, to later automatically locate appropriate connecting streams and subgraphs. Table 11 shows the number of cutsets and the connecting streams at various locations of a single decomposition. We solve the problem with single decomposition at the location with connecting streams S_{18} and S_8 which gives the minimum number of cutsets.

We now perform a higher number of decompositions (more than one) on the flow diagram, and for every decomposition the location at which the total number of cutsets are minimum, we solve the problem. Table 12 shows the computation time, nodes explored, total cutsets, and connecting streams for each run.

We found the computation time increasing with the number of decompositions, and therefore, we stop at five decompositions. All the runs were executed on an Intel Celeron 1.39 GHz, 256 MB RAM PC, and all gave the same solution with a cost of 185 and measuring streams $\{S_3, S_4, S_6, S_7, S_8, S_9, S_{10}, S_{12}, S_{13}, S_{16}, S_{17}, S_{19}, S_{20}, S_{23}, S_{24}\}$, $\{S_3, S_4, S_6, S_7, S_8, S_9, S_{10}, S_{12}, S_{14}, S_{16}, S_{17}, S_{19}, S_{20}, S_{23}, S_{24}\}$. Table 12 shows the decrease in number of cutsets and increase in number of connecting streams with the increase in the number of decompositions.

A node counter is used in the cutset tree algorithm which is incremented whenever the node is evaluated with the model (either feasible or not); the optimal node is the node counter number at which the tree finds the optimal solution. The optimal node value gives an idea of how early the optimal solution is found in the tree. The earlier the optimal node is found, the tighter the lower bound for the stopping criteria, avoiding deeper exploration of the tree. The total nodes explored is not the measure of computational speed; for example, more time (2 min 25 s) was required in 5 decompositions to explore 10 898 nodes while less time (43 s) was required in 2 decompositions to explore 10 462 nodes. This is because the number of single node evaluations due to different combinations of union and ring sum operations (for connecting streams) is different and hence the average computation time of a node for different runs is different.

The computation time is the smallest for the run with two decompositions at streams $\{S_{16}\}$, $\{S_{10}, S_8\}$ and, then, increases with the number of decompositions. We fix this computation time as the lower bound time and run the Madron program for two decompositions at different locations around $\{S_{16}\}$, $\{S_{10}, S_8\}$. Any run with computation time greater than 43 s is terminated. Similar runs are also performed for single and triple decompositions and around the locations $\{S_{18}, S_8\}$ and $\{S_{15}\}$, $\{S_{18}, S_8\}$, and $\{S_3\}$, respectively. No other runs gave the solution with a computation time less than 43 s.

Example 3. This example is considered to illustrate the efficiency of the method. A flow sheet of the crude distillation unit (CDU) shown in Figure 8 is taken from Bagajewicz et al.[5] Vapor and mixed flow streams were eliminated. Figure 8 consists of 52 streams and 19 units. Table 13 and Table 14 contain the considered values of mass flow and sensor costs, respectively. All sensors are available at 2% precision. A precision of 5% is required in streams S_{17}, S_{18}, S_{31}, S_{33}, S_{35}, S_{37}, S_{43}, S_{44}, S_{45}, S_{46}, S_{47}, S_{48}, S_{50}, S_{51}, and S_{52} ($\sigma_i^* = 5\%$, $i = 17, 18, 31, 33, 35, 37, 43, 44, 45, 46, 47, 48, 50, 51, 52$). A residual precision order of a and a threshold of 25% in the same variables ($\psi_i^* = 25\%$, $i = 17, 18, 31, 33, 35, 37, 43, 44, 45, 46, 47, 48, 50, 51, 52$) are required. Also, error detectability of $\kappa^{\mathrm{D}} = 3.9$ (with $\gamma = 50\%$) for flow 31, 37, and 43 was added.

We start with finding the number of cutsets for single decomposition at different locations and solve the CDU problem at the location which gives least number of cutsets (the appendix shows the method of finding optimal locations). Table 15 shows the number of cutsets for single decomposition of the CDU at different locations along the length. A single decomposition with the connecting stream $\{S_{29}\}$ gives the least number of cutsets. Similarly, we find the locations for higher numbers of decompositions, which gives the least number of cutsets. For each of these locations, we solve the CDU problem, and Table 16 shows the results obtained. If we find more than one location with same number of reduced cutsets, we solve the CDU problem at all those locations.

All the runs were executed using an Intel Celeron 1.39 GHz, 256 MB RAM PC, and all gave the same solution with a cost of 44 100 measuring the following streams $\{S_{15}, S_{16}, S_{17}, S_{18}, S_{19}, S_{29}, S_{31}, S_{32}, S_{33}, S_{35}, S_{37}, S_{42}, S_{43}, S_{44}, S_{45}, S_{46}, S_{47}, S_{48}, S_{49}, S_{50}, S_{51}, S_{52}\}$ or $\{S_{15}, S_{16}, S_{17}, S_{18}, S_{19}, S_{29}, S_{31}, S_{32}, S_{33}, S_{35}, S_{36}, S_{42}, S_{43}, S_{44}, S_{45}, S_{46}, S_{47}, S_{48}, S_{49}, S_{50}, S_{51}, S_{52}\}$. The computational time highly depends on the number of connecting streams, their cost, and how early the optimal node is found. For example, a decomposition in three subgraphs at streams

Table 18. Results for Example 3 for Two Decompositions at Different Locations

no. of subgraphs	total cutsets	connecting streams	connecting streams cost	time	nodes explored	optimal node
1	188	$\{S_{15}\}$	1500	111 min 32 s	518 234	865
1	248	$\{S_{13},S_{14}\}$	3600	183 min 23 s	712 345	4387
1	310	$\{S_{12}\}$	2000	135 min 42 s	603 723	3128
3	74	$\{S_{11},S_{15},S_{29}\}$	5000	5 min 34 s	5 581	431
3	74	$\{S_{5},S_{15},S_{29}\}$	5200	5 min 47 s	5 723	314
3	74	$\{S_{15},S_{29},S_{42}\}$	5300	5 min 50 s	5 821	257
3	74	$\{S_{5},S_{12},S_{29}\}$	5700	6 min 41 s	7 990	856
3	76	$\{S_{11},S_{29},S_{49}\}$	5800	8 min 2 s	12 734	2277
3	76	$\{S_{12},S_{29},S_{49}\}$	5800	7 min 21 s	11 504	1264
3	77	$\{S_{5},S_{11},S_{29}\}$	5700	7 min 48 s	12 338	1453
3	78	$\{S_{12},S_{15},S_{29}\}$	5000	5 min 54 s	5 987	182
3	80	$\{S_{2},S_{12},S_{29}\}$	5500	6 min 18 s	7 325	314
3	80	$\{S_{11},S_{12},S_{29}\}$	5500	6 min 55 s	8 156	465
3	80	$\{S_{5},S_{29},S_{42}\}$	6000	7 min 3 s	8 442	342

$\{S_{12}\}$,$\{S_{29}\}$, and $\{S_{42}\}$ required 8 min 21 s to explore a total of 10 653 nodes (with optimal node 108) while a decomposition into two subgraphs at streams $\{S_{12}\}$ and $\{S_{29}\}$ required less time, 5 min 4 s, to explore more nodes 12 990 (with the optimal node being 856). Although the number of nodes explored is higher, there are a lower number of union-ring sum operations at each node, thus requiring less time. These operations are not performed on every node of the tree: only when they contain the connecting streams. Hence, the frequency of getting a node with connecting streams changes for different numbers of decompositions and their locations.

The run with 2 decompositions at streams $\{S_{12}\}$, $\{S_{29}\}$ gave the minimum computational time and hence formed the new lower computational bound. With this lower computational bound of 5 min 4 s, we performed more runs on the CDU example with two decompositions at other locations from streams S_{12} and S_{29}. These runs are shown in Table 17. Although these runs are longer, they were not interrupted at the lower computational bound but were allowed to complete for illustration purposes.

The runs with the locations having many connecting streams, $\{S_{12}\}$, $\{S_{20}-S_{28}\}$ and $\{S_{12}\}$, $\{S_{38}-S_{41}\}$, take a very long time. This is due to a large number of union-ring sum operations at each node and also because of a large number of connecting stream costs which loosens the lower bound stopping criteria and forces a deeper exploration of the tree. Hence, we avoid executing the runs at the locations with many connecting streams such as $\{S_{20}-S_{28}\}$ and $\{S_{38}-S_{41}\}$. A run with a computation time smaller than the lower time bound was obtained at location $\{S_{15}\}$, $\{S_{29}\}$.

Similar runs with single decomposition with locations varying along stream $\{S_{29}\}$ and triple decompositions with locations varying along $\{S_{11}\}$,$\{S_{29}\}$,$\{S_{42}\}$ were performed. All the runs gave the same solution. No other runs with times lower than 4 min 12 s were obtained, and Table 18 shows these results.

The CDU was decomposed into the maximum number of decompositions which is 19; all the streams except the one connected to the environment are connecting streams. The algorithm explored a total of 18 850 nodes in 3 days and had skipped 1247 nodes when it was interrupted. Assuming the same fraction of nodes to be explored from remaining nodes, the estimated time of completion is 83 days. Due to the large number of connecting streams, large numbers of union-ring sum combinations are performed among them and hence the average computation time for a single node is very large. In other words, it would perform even worse than using a union of cutsets.

Conclusions

We presented a decomposition algorithm to be used in a tree of cutsets that reduces considerably the computational time needed to solve large industrial size problems. The algorithm is rigorous, that is, it guarantees global optimality. An example shows that, for an industrial size flow sheet, the solution time can be reduced from an estimated 45 days to 4 min.

Appendix

Systematic Approach for Identifying Sub-graphs. Identifying appropriate connecting streams for decomposition can be

$$\begin{array}{c} \\ U_1 \\ U_2 \\ U_3 \\ U_4 \end{array}\begin{array}{c} \begin{array}{ccccccc} S_1 & S_2 & S_3 & S_4 & S_5 & S_6 & S_7 \end{array} \\ \left[\begin{array}{ccccccc} 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & -1 \end{array}\right] \end{array}$$

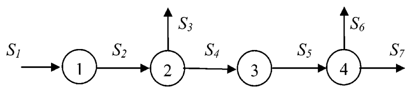

Figure A1. Seven stream flow diagram.

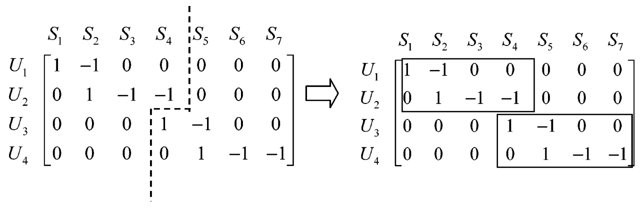

Figure A2. Single decomposition for Figure A1.

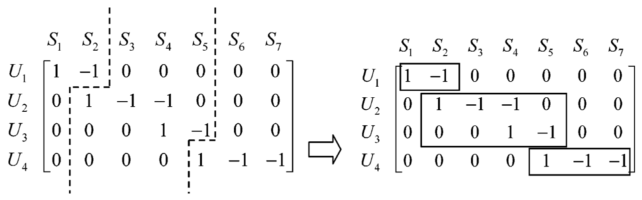

Figure A3. Double decomposition for Figure A1.

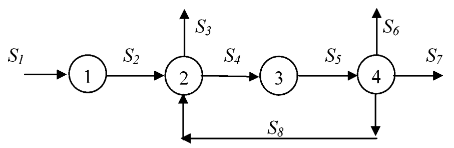

Figure A4. Eight stream flow diagram.

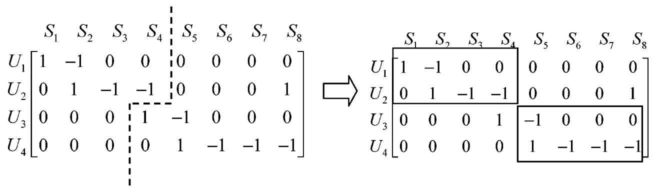

Figure A5. Single decomposition for Figure A3.

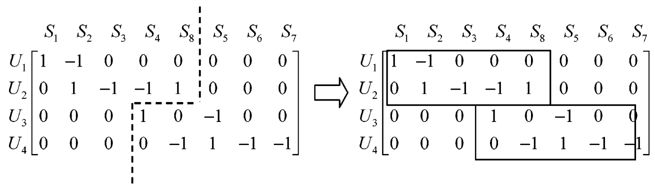

Figure A6. Single decomposition for Figure A3, rearranged.

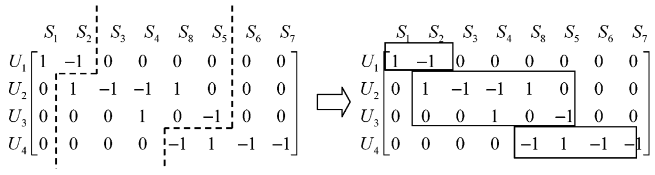

Figure A7. Double decomposition for Figure A3.

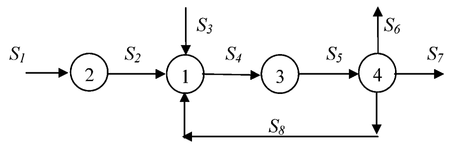

Figure A8. Two input flow diagram.

$$\begin{array}{c|cccccccc} & S_1 & S_2 & S_3 & S_4 & S_5 & S_6 & S_7 & S_8 \\ \hline U_1 & 0 & 1 & 1 & -1 & 0 & 0 & 0 & 1 \\ U_2 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ U_3 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ U_4 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & -1 \end{array}$$

Figure A9. Incidence matrix for Figure A8.

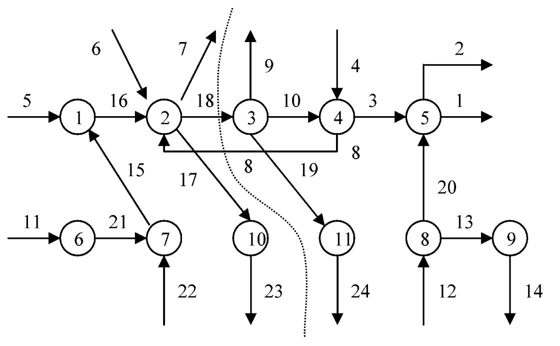

Figure A10. Illustration of single decomposition in streams S_{18} and S_8.

done by simple inspection, especially if a graph is linear like the one in Figure 8. Even in cases such as the one in Figure 7, one could identify places where the graph is divided into equal parts. We want, however, to be able to do this automatically. In this appendix, we show how this can be done.

The incidence matrix for any process flow diagram is constructed as follows:

(1) Each row of the incidence matrix represents a unit and each column represents a stream.

(2) In each row, streams entering the corresponding unit are represented by a number 1. Streams leaving the unit are represented by a number −1.

Consider the graph of Figure A1, where the incidence matrix is shown.

For linear flow diagrams (Figure A1), if the first stream in the incidence matrix corresponds to a feed stream, then the elements of the incidence matrix along the diagonal are dense (nonzero with 1 and −1) while the elements in the nondiagonal region are zero. To be able to find the best decomposition into two subgraphs, we aim at leaving equal or roughly equal number of nodes in each subgraph. To do this, we simply choose to divide the incidence matrix into several matrices, all of them with the same number of rows. For example, for a single decomposition, we decompose the incidence matrix below row U_2. This leads us to identify column S_4 (Figure A2), which results in single decomposition with connecting stream S_4. The dotted line shows the breakage, which leaves S_4 as an exit stream for the matrix on the left and an input stream for the matrix on the right. Figure A2 also shows the resulting matrices, which, for S_4 to be the only connecting stream, must fulfill two properties, (a) the resulting subgraphs need to be independent and (b) the resulting subgraphs need to be connected.

For the graphs to be independent, no stream, except the connecting stream, should be in both graphs. This can be easily checked in the example of Figure A2 by verifying that the rows below row U_2 below the left matrix and the rows above U_3 above the right matrix have all zero elements. One now needs to check that the subgraphs generated are connected.

The method to determine connectivity based on the incidence matrix is as follows (see chapter 5 of the work of Bagajewicz[3]):

(1) Take the first row of the incidence matrix and find a column with a nonzero element.

(2) In that column, search for another row with a nonzero element, and by construction, this element will have an opposite sign.

(3) Add the latest row found to the first row and eliminate the row and the column.

(4) Repeat the same procedure for all the remaining rows.

(5) Finally, count the number of rows left. This number is the number of connected systems.

For double decomposition, we decompose the incidence matrix below row U_1 and below row U_3 (shown in Figure A3) which results in double decomposition with connecting stream $\{S_2\}$ and $\{S_5\}$.

We now illustrate the procedure using slightly more complex flowsheets. Consider now a linear flow diagram with a recycle stream as shown in Figure A4. Figure A5 shows the single

$$\begin{array}{c|cccccccccccccccccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 \\ \hline U_1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ U_2 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ U_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ U_4 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ U_5 & -1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ U_6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ U_7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ U_8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ U_9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ U_{10} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ U_{11} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 \end{array}$$

Figure A11. Incidence matrix for Figure A10.

$$\begin{array}{c|cccccccccccccccccccccccc}
 & 11 & 21 & 22 & 15 & 5 & 16 & 6 & 7 & 17 & 23 & 8 & 18 & 9 & 19 & 24 & 10 & 4 & 3 & 1 & 2 & 20 & 12 & 13 & 14 \\
\hline
U_6 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_7 & 0 & 1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_1 & 0 & 0 & 0 & 1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_2 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_{10} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_{11} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & 1 & 0 & 0 & 0 \\
U_8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & -1 & 0 \\
U_9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1
\end{array}$$

Figure A12. Diagonal incidence matrix for Figure A10.

$$\begin{array}{c|cccccccccccccccccccccccc}
 & 11 & 21 & 22 & 15 & 5 & 16 & 6 & 7 & 17 & 23 & 8 & 18 & 9 & 19 & 24 & 10 & 4 & 3 & 1 & 2 & 20 & 12 & 13 & 14 \\
\hline
U_6 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_7 & 0 & 1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_1 & 0 & 0 & 0 & 1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_2 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_{10} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_{11} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & 1 & 0 & 0 & 0 \\
U_8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & -1 & 0 \\
U_9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1
\end{array}$$

Figure A13. Single decomposition of Figure 7.

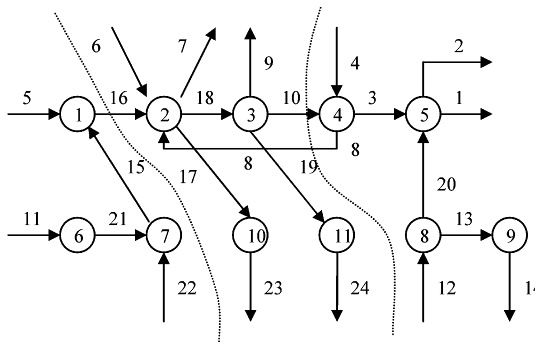

Figure A14. Illustration of a double decomposition in streams S_{16} and $\{S_8, S_{10}\}$.

decomposition. We first notice that the incidence matrices for the subgraphs are not independent. Stream S_8 is present in both decomposed incidence matrices and therefore needs to be added as a connecting stream. We therefore rearrange the matrix to put stream S_8 closer to S_4 and proceed to partition through both streams. This is shown in Figure A6.

For double decomposition, we decompose the incidence matrix below row U_1 and below row U_3 (shown in Figure A7) which results in double decomposition with connecting stream $\{S_2\}$ and $\{S_5, S_8\}$.

We now consider the situation of graphs that contain more than one feed stream. Here, the question arises as to which feed streams are appropriate to start. We first notice that we prefer the streams that are feeding units not connected to any recycle stream or connected to streams that are coming from other nodes. For the flow diagram shown in Figure A8, we have two input streams S_1 and S_3. Figure A9 shows the incidence matrix for Figure A8. Two input streams make the incidence matrix nondiagonal. Hence, we use the row and column interchange operations to get a diagonal incidence matrix. The proposed procedure to obtain an all-diagonal incidence matrix is as follows:

(1) We consider all input and output streams of the process flow diagram and select only the input streams connected to a node which do not get any input from another node. We do the same with output streams.

(2) We first start with the selected input streams. By performing the row and column exchange operations, we get the selected input stream as the first element of the incidence matrix (first row and first column).

$$\begin{array}{c|cccccccccccccccccccccccc}
 & 11 & 21 & 22 & 15 & 5 & 16 & 6 & 7 & 17 & 23 & 24 & 18 & 9 & 19 & 8 & 10 & 4 & 3 & 1 & 2 & 20 & 12 & 13 & 14 \\
\hline
U_6 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_7 & 0 & 1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_1 & 0 & 0 & 0 & 1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_2 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_{10} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_{11} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
U_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & 1 & 0 & 0 & 0 \\
U_8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & -1 & 0 \\
U_9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1
\end{array}$$

Figure A15. Double decomposition of Figure A14.

(3) Now by performing the column exchange operations, we get the rest of the streams present in the same row (same unit) closer to the selected input stream. Streams with 1 are placed before streams with −1.

(4) We now find a row (from the remaining rows) which has at least one stream being the same as the previous row streams and place that row below the first row (by row interchange operations). Again, we perform the column exchange operations as described in step 3.

(5) Continue with steps 4 and 5 until the last row is reached.

The same procedure is performed for all selected inputs. For selected output streams, we placed them at the last row and last column and performed the same procedure in a reverse way until the first row of the incidence matrix is reached. We now illustrate this.

Consider the flowsheet of Figure A10 and the proposed decomposition. Its incidence matrix is shown in Figure A11.

The incidence matrix is not diagonally dense; hence, we convert the incidence matrix using the input stream S_{11} as the selected input (Figure A12).

For single decomposition, we decompose the incidence matrix below row U_{10} which leads in almost two equal parts. This leads us to identify column S_{18} and S_8 (Figure A13).

Figure A14 shows the same flowsheet with a proposed double decomposition with connecting streams $\{S_{16}\}$ and $\{S_8, S_{10}\}$. The corresponding incidence matrix is shown in Figure A15.

Literature Cited

(1) Gala M.; Bagajewicz, M. Rigorous Methodology for The Design and Upgrade of Sensor Networks using Cutsets. *Ind. Eng. Chem. Res.* **2006**, 6679−6686.

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

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

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

(5) Bagajewicz, M.; Markowski, M.; Budek, A. Economic Value of Precision in the Monitoring of Linear Systems. *AIChE J.* **2005**, *51* (4), 1304.

Received for review July 25, 2005
Revised manuscript received July 8, 2006
Accepted July 11, 2006

IE050866J