# Topics in Computer-Aided Design:

## Part I. An Optimum Tearing Algorithm for Recycle Systems

The concepts of ineligible streams and two-way edge reduction are extended to simplify the signal flow diagram of a recycle process flow sheet graph. The solution to a tearing problem can be obtained readily by repeated reduction of ineligible streams and two-way edges. When this fails, a branch and bound method will guarantee optimality at the expense of a few enumerations.

T. K. PHO
and L. LAPIDUS
Department of Chemical Engineering
Princeton University
Princeton, New Jersey 08540

## SCOPE

One approach to the design and optimization of a large processing system is to develop an executive computer program coordinating the subroutines which perform the design computation for the process units. Integral in this approach is the specification, a priori, of a precedence-ordering which determines the sequence in which each subroutine is to be computed. An input process stream to a subroutine which has not been specified at the time of computation is called a recycle or torn stream because the stream must be cut by assuming initial values to all the variables that are present in that stream. This process allows the subroutine computation to be continued. An iterative method is then employed to force convergence of the torn stream to within a specified tolerance.

The present paper solves the minimum weighted recycle stream problem where each stream is assigned an arbitrary weighting factor, and a precedence-ordering is sought which gives rise to a set of torn streams with minimum weighting sum. This approach includes both the minimum recycle variable and the minimum recycle stream problems as special cases. The first is obtained by equating the weighting factor to the number of variables that are present in a stream, and the second by allowing all the weightings to be unity.

Two previous algorithms using the method of dynamic programing (Sargent and Westerberg, 1964; Upadhye and Grens, 1972) have been proposed which can always obtain an optimal torn set. Because of the inherent dimensionality difficulty associated with dynamic programing, these methods cannot, in general, be used to solve large recycle problems. A more efficient procedure, which however does not always guarantee optimality, is the direct graph simplification technique first proposed by Sargent and Westerberg (1964) and later expanded by Christensen and Rudd (1969). The simplification technique seeks to reduce a process flow sheet graph to a null graph by systematically eliminating streams and merging nodes in a manner such that the problem solution is unchanged. The simplifications involve (1) the two-way edge reduction which cuts a recycle loop formed by two process streams and (2) the elimination of process streams which are ineligible to be a member of the optimal torn set. In many cases considered, the simplification technique results in a residual graph which must be further analyzed by other combinatorial techniques. Sargent and Westerberg used the dynamic programing approach, and Christensen and Rudd suggested examining all the possible combinations of the index nodes. Both approaches are not always efficient and can involve severe combinatorial difficulty. Barkley and Motard (1972) suggested a different variation to the graph simplification technique by performing an interval reduction procedure to a signal flow diagram which is readily obtainable from the process flow sheet graph. This method, however, is limited to problems with only unit weightings.

In this paper we shall derive a Basic Tearing Algorithm (BTA) which is shown to be a generalization on the previous concepts of ineligible stream and interval reduction. The BTA identifies and eliminates the ineligible streams directly on a signal flow diagram, thus achieving a reduction of the problem solution. The concept of two-way edge reduction will also be extended to the signal flow diagram to aid the effectiveness of the BTA. Any residual graph which is not reducible by the BTA and the two-way edge reduction can be further analyzed by a branch and bound method. Detailed examples to illustrate the methods are given.

## CONCLUSIONS AND SIGNIFICANCE

The Basic Tearing Algorithm is shown to be a generalization of the previous concepts of ineligible streams which can lead to the identification of a greater number of the ineligible streams. A systematic elimination of these streams from a signal flow diagram will lead eventually to the identification of the optimal torn set. It is further shown that the concept of interval reduction is a special case of the BTA when the latter is applied to a recycle problem with unit weightings. The algorithm here proposed to ensure complete reduction of a recycle problem consists of first using the BTA, then the two-way edge reduction, and finally a branch and bound method. The first two parts are found to be sufficient to solve a large class of problems. This is illustrated by applying the method to a number of well-known examples, as well as counter-examples which previous methods have failed to resolve. In the event that the first two parts of the algorithm fail to reduce a signal flow diagram completely, the branch and bound method guarantees an optimal solution at the expense of few additional enumerations.

## GRAPH THEORY

As we shall repeatedly refer to the properties of a finite directed graph, a brief review of graph theory is in order.

A finite directed graph $G(V, E)$ is a finite collection of nodes $V$ whose elements are joined by a finite set of directed edges $E$. Two nodes $v_i$ and $v_j$ from $V$ are joined by a directed edge $e_{ij}$ if and only if the edge $e_{ij}$ is originating from $v_i$ and terminating on $v_j$. The node $v_i$ is referred to as being an immediate predecessor of $v_j$, and $v_j$ as an immediate successor of $v_i$. The set of all the immediate successors to a node $v$ is called the mapping of $v$ by a function $T$ and is denoted by $T(v)$. Conversely, the set of all the immediate predecessors to $v$ is denoted by the inverse mapping $T^{-1}(v)$.

A path between two nodes $v_{i_0}$ and $v_{i_n}$ in $G(V, E)$ is an ordered collection of directed edges $e_{i_0 i_1} e_{i_1 i_2} \ldots e_{i_{n-1} i_n}$ such that $e_{i_0 i_1}$ originates from $v_{i_0}$, and $e_{i_{n-1} i_n}$ terminates on $v_{i_n}$. As each edge is directed toward a node, a path is called simple if it encounters no node twice and cyclic if it originates and terminates on the same node, that is, $i_0 = i_n$. A simple loop is a simple cyclic path without regard to its endpoints.

A cyclical graph is a subset of $G(V, E)$ such that there exists a simple path within the set from any node to another node. It can be shown easily that a cyclical graph contains at least a simple loop. A cyclical loop is maximal if and only if it is cyclical and contains all other cyclical graphs as its subgraph. If a graph contains no simple loop, it is called acyclic. The in-degree $\delta^+(v)$ of a node $v$ in $G(V, E)$ is the number of edges directed toward it, and its out-degree $\delta^-(v)$ is the number of edges directed out from $v$. The sum of in- and out-degrees is called the degree of $v$ and is denoted by $\delta(v)$.

## PRINCIPLE OF DECOMPOSITION

A method of representing a process plant for computer simulation is to construct a process flow sheet graph in which the nodes represent the subroutines for unit computations and the directed edges represent the process streams. Each node therefore defines a mapping whereby the variables in the output streams can be computed by specifying the variables in the input streams. As a rule, only the interstreams between the nodes are retained while the interface feed and product streams are deleted from the graph. The result obtained is a finite directed graph.

The simulation begins by first specifying a precedence-ordering which determines the computation sequence of the process unit subroutines. If an input stream to a node is not specified at the time of computation, it is called a recycle stream. It can be shown that for an acyclic graph a precedence-ordering can be found so that it gives rise to no recycle streams (Christensen and Rudd, 1969). The same method used to determine this precedence-ordering can also be used to delete all the acyclic subgraphs from the original flow sheet graph, leaving a cyclic subgraph. A useful concept which can reduce the labor of analyzing a complex cyclic graph is the partitioning of the graph into a number of maximal subgraphs. This enables the recycle calculations to occur within a smaller subgraph than the original. The maximal cyclical graphs can be further ordered so that completing the computations in one subgraph will assure that all the variables in the subsequent subgraphs be known. We shall henceforth refer to a recycle system as being one of the maximal subgraphs. Methods of performing the above partition have been described elsewhere (Christensen and Rudd, 1969; Ledet and Himmelblau, 1970) and are omitted here.

### Tearing

By definition of the maximal cyclical graph, a recycle system must contain at least a simple loop. A common method of cutting this loop is to assume initial values for all the variables in any one of the streams which constitute the loop. The loop is said to be torn at the chosen stream. If all the loops in the recycle system are torn in this manner, the resulting graph becomes acyclic and can then be precedence-ordered and computed to produce a set of new values for the torn variables. An iteration procedure is then performed to force the agreement between the assumed and computed torn variables to some specified tolerance.

It is apparent that different combinations of the streams can be chosen as the torn set, and the set with the minimum number of variables would appear as a good choice since it gives rise to the fewest iterates. A method by which this torn set can be selected is to assign a weighting factor to each process stream and to set the value equal to the number of variables that are present in that stream. A learning procedure can be used to adjust the weighting of any stream which converges with difficulty (Sargent and Westerberg, 1964). Therefore we can view the optimum tearing problem as being directed to finding a torn set with a minimum weighting sum.

## THEORY: A LOOP MATRIX FORMULATION

Let $L = \{l_i\}$ denote the set of all the simple loops in a recycle system which is represented by its finite directed graph $G = G(V, E)$. Methods of tracing all the loops have been described by Norman (1965) and Weinblatt (1972). We can define a loop matrix $A$ whose elements are given by

$$a_{ij} \begin{cases} 1, & \text{if stream } s_j \text{ formed part of loop } l_i \\ 0, & \text{otherwise} \end{cases}$$

Let us further define $R_i = \{s_j | a_{ij} = 1\}$ representing the set of stream columns whose row $i$ have nonzero element 1 and $C_j = \{l_i | a_{ij} = 1\}$ representing the set of loop rows whose column $j$ have nonzero element 1. Clearly all the loops in the set $C_j$ can be torn by selecting the stream $s_j$ as the iterate. Stated in a different way, the column $s_j$ is said to cover the set of rows in $C_j$. A set of stream columns which cover all the rows in the loop matrix will therefore open all the loops in $G$ and is a feasible torn set. The set which gives rise to the minimum weighting sum can be found by solving the following linear integer programming problem:

$$\min_{\{x_j\}} \sum_{j=1}^{N} p(s_j)x_j$$

subject to

$$\sum_{j=1}^{N} a_{ij}x_j \geq 1 \quad i = 1, 2, \ldots, M \tag{1}$$

and

$$x_j = 1 \text{ or } 0$$

where $p(s_j)$ is the weighting factor of stream $s_j$, $N$ is the number of streams in $G$, and $M$ is the number of loops in $L$. A stream $s_j$ is selected as a torn stream if $x_j = 1$, otherwise $x_j = 0$. The first constraint restricts the selection of the torn set so that all the rows in the loop matrix can be covered. The solution of (1), given the matrix $A$, is also known as a minimum cover problem which finds applications in fields other than recycle problems (Roth, 1969; Rubin, 1971). An analytic solution to (1) is, at present, not feasible when the size of the loop matrix is moderately large. Two tearing algorithms which have used the loop matrix to solve (1) in a different manner have appeared in the chemical engineering literature (Lee and Rudd, 1966; Upadhye and Grens, 1972). However, the use of their methods must be preceded by finding the loop matrix. In this paper, we shall utilize some results from the covering problem to develop a new tearing algorithm. The final algorithm does not require knowledge of the loop matrix as in the previous work.

The following reduction procedures and definitions are well known in the solution of the covering problem (Roth, 1969):

(i) Row dominance: If there exist two rows $i$ and $j$ such that $R_i \subseteq R_j$, then any stream column that covers row $i$ must also cover row $j$; thus row $j$ can be deleted from matrix $A$ without affecting the optimality of (1).

(*ii*) Column dominance: If there exists a stream column $s_m$ and a stream index set $\Gamma$ such that

$$C_m \subseteq \bigcup_{k \epsilon \Gamma} C_k$$

and

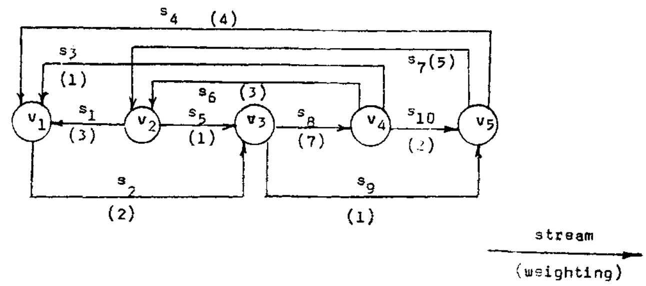
$$p(s_m) \geq \sum_{k \epsilon \Gamma} p(s_k) \tag{2}$$

are satisfied, then the stream set $\{s_k | k \epsilon \Gamma\}$ are said to dominate the stream column $s_m$. This is because the rows that can be covered by $s_m$ can also be covered more effectively by the first stream set with lesser cost. The stream column $s_m$ is therefore redundant in the presence of stream columns whose indices are in $\Gamma$ and can be deleted from $A$.

(iii) Essential column: A stream column $s_m$ is essential if and only if there is a row $i$ whose only nonzero element lies in the same column, that is, $R_i = \{s_m\}$. In this case the only way that row $i$ can be covered is by selecting $s_m$ as being a member of the optimal solution. Since selecting $s_m$ also cuts all the loops in $C_m$, we can delete both the stream column $s_m$ and all the rows in $C_m$ from the loop matrix.

The essential columns usually appear after the row and column dominances are repeatedly identified and eliminated. If the reduction process can continue until all the rows have been covered, an optimal solution to (1) is obtained without resorting to an analytical method.

To illustrate the procedures, consider the recycle system shown in Figure 1a with the weighting factors enclosed in the parenthesis. The loop matrix to the system is shown in Figure 1b. In the matrix, $C_1 = \{l_1, l_5, l_7\} \subset C_2 =$




(a)

| | $s_1$ | $s_2$ | $s_3$ | $s_4$ | $s_5$ | $s_6$ | $s_7$ | $s_8$ | $s_9$ | $s_{10}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $l_1$ | 1 | 1 | | | | | 1 | 1 | | 1 |
| $l_2$ | | | | | 1 | | 1 | 1 | | 1 |
| $l_3$ | | 1 | | 1 | | | | 1 | | 1 |
| $l_4$ | | 1 | 1 | | | | | 1 | | |
| $l_5$ | 1 | 1 | | | | 1 | | 1 | | |
| $l_6$ | | | | | 1 | 1 | | 1 | | |
| $l_7$ | | 1 | | 1 | | | | | 1 | |
| $l_8$ | 1 | 1 | | | | | 1 | | 1 | |
| $l_9$ | | | | | 1 | | 1 | | 1 | |
| $p(s_i)$ | 3 | 2 | 1 | 4 | 1 | 3 | 5 | 7 | 1 | 2 |

(b)

**Fig. 1. A recycle system and its loop matrix.**

$\{l_1, l_3, l_4, l_5, l_7, l_8\}$ and $p(s_1) > p(s_2)$; thus stream column $s_1$ is dominated by $s_2$ and is deleted from $A$. Similarly, all the stream columns $s_6$, $s_7$, and $s_8$ are dominated by the stream set $\{s_2, s_5\}$ and their removal results in essential columns $s_2$ and $s_5$ appearing in rows 5 and 6, respectively. The deletion of both the essential columns and all the rows in $C_2$ and $C_5$ will cut all the loops in the recycle system. Thus, the optimal torn set has been found to be $\{s_2, s_5\}$.

We note here that the same line of reasoning is also used in the tearing algorithm proposed by Lee and Rudd (1966). When the loop matrix has a large number of stream columns, it is apparent that the checking for containment which satisfies the column dominance in (2) can involve severe combinatorial difficulty. This in turn makes the solution by this approach less appealing. We shall later derive a more direct method of using the column dominance.

### Two-way Edge Reduction

Although the row and column dominances and the essential column can reduce the problem solution without sacrificing optimality, they do not always reduce the loop matrix completely. A two-way edge reduction can be derived to further reduce the matrix.

Let us consider a row $p$ from a loop matrix which has only two nonzero elements in the stream columns $s_m$ and $s_n$, that is, $R_p = \{s_m, s_n\}$. The loop $l_p$ is said to form a two-way edge between streams $s_m$ and $s_n$. A two-way edge is simple if and only if it does share $s_m$ and $s_n$ with other two-way edges. In order to cut loop $l_p$, it is necessary and sufficient to select only one stream from $R_p$ to be the torn stream and never both. This follows from the obvious fact that in any precedence-orderings involving nodes $v_m$ and $v_n$ where $s_m$ and $s_n$ are their output streams, respectively, only one stream from $R_p$ can serve as a recycle stream. For example, if $v_m$ proceeds $v_n$ in the precedence-ordering, then $s_n$ will be the recycle stream, and vice versa.

Without loss of generality, let us assume $p(s_m) \leq p(s_n)$ where $p(s_m)$ and $p(s_n)$ are the weighting factors of streams $s_m$ and $s_n$, respectively. Subtract $p(s_m)$ from the weightings of both $s_m$ and $s_n$, and call the solution to (1), with these new modified weightings, problem (1′). It follows that the optimal torn set obtained from solving (1′) will be the same as (1), except that its optimal objective function will be smaller by the amount $p(s_m)$. In solving problem (1′), we note that at least one stream $s_m$ will have a zero weighting factor and will not contribute to the optimal objective function. We can therefore regard $s_m$ as being a probable torn stream and proceed to delete stream column $s_m$ and loop rows in $C_m$ from the modified loop matrix. This can result in a reduced loop matrix from which the reductions of row and column dominances and essential stream can be further employed.

A probable torn stream will become a torn stream provided that the compliment stream $s_n$ which forms a simple two-way edge with $s_m$ is later found to be dominated by another stream set; otherwise the probably torn stream is deleted from the optimal torn set. This is because only one stream from the set $\{s_m, s_n\}$ can be chosen as the torn stream.

#### Non-Simple Two-Way Edges

A two-way edge between two streams $s_m$ and $s_{n_1}$ is non-simple if at least one of the streams is shared by another two-way edge. The non-simple two-way edges arise frequently when a loop matrix is sufficiently reduced by the row and column dominances. Let us assume $s_m$ also forms a two-way edge between $s_{n_2}, s_{n_3}, \ldots, s_{n_q}$, respectively. In order to cut all the loops formed by the two-way edges between $s_m$ and the stream set $S_N \triangleq \{s_{n_1}, s_{n_2}, \ldots, s_{n_q}\}$, it is necessary and sufficient to tear at either $s_m$ or the stream set $S_N$, and never both. Therefore, if any stream from $S_N$ is dominated by another stream set, all the streams in $S_N$ must be taken as being dominated streams, and if any stream is essential, all are also essential. This follows from the fact that if all the streams but $s_p$ in $S_N$ are torn streams, then in order to cut the two-way edge between $s_m$ and $s_p$, the stream $s_m$ must be taken as a torn stream. This, however, is impossible, since all streams which also form a two-way edge with $s_m$ have been taken as torn streams, and this automatically excludes $s_m$ from being a torn stream.

In view of this result, we can rename all the stream columns $s_{n_1}, s_{n_2}, \ldots, s_{n_q}$ by their set name $S_N$, and compute the weighting of $S_N$ as the weighting sum of all streams in $S_N$. Note that there are now $q$ columns with name $S_N$ and weighting $p(S_N)$. We can now regard a two-way edge as being formed between stream column $s_m$ and any column named $S_N$ and proceed to apply the previous two-way edge reductions. If $p(S_N) \leq p(s_m)$, then all the $q$ columns named $S_N$ are declared as probably torn streams; otherwise $s_m$ is a probable torn stream.

As an example of two-way edge reduction, consider the loop matrix shown in Figure 2a. No row and column dominances can be identified, but all the loops are observed to be formed by two-way edges. Since the stream $s_1$ forms two-way edges with streams $s_2$, $s_4$, $s_5$ and $s_6$, we have $S_N = \{s_2, s_4, s_5, s_6\}$ and $p(S_N) = 1 + 3 + 1 + 2 = 7$ (Figure 2b). Since $p(S_N) > p(s_1)$, the stream $s_1$ is a probable torn stream; its removal results in the reduced matrix shown in Figure 2c with the weighting $p(S_N)$ reduced by 2. In the reduced matrix, column 3 is dominated by stream column $s_3$; its deletion leaves $s_3$ as essential. This step also implies that the probable torn stream $s_1$ belongs to the optimal torn set. Deletion of the essential stream $s_3$ will cut all the remaining loops, thus an optimal torn set has been found to be $\{s_1, s_3\}$.

| | $s_1$ | $s_2$ | $s_3$ | $s_4$ | $s_5$ | $s_6$ |
|---|---|---|---|---|---|---|
| $l_1$ | 1 | | | | | 1 |
| $l_2$ | | | 1 | | | 1 |
| $l_3$ | 1 | 1 | | | | |
| $l_4$ | 1 | | | 1 | | |
| $l_5$ | | | 1 | 1 | | |
| $l_6$ | 1 | | | | 1 | |
| $l_7$ | | | 1 | | 1 | |
| $p(s_j)$ | 2 | 1 | 2 | 3 | 1 | 2 |

(a)

| | $s_1$ | $S_N$ | $s_3$ | $S_N$ | $S_N$ | $S_N$ |
|---|---|---|---|---|---|---|
| $l_1$ | 1 | | | | | 1 |
| $l_2$ | | | 1 | | | 1 |
| $l_3$ | 1 | 1 | | | | |
| $l_4$ | 1 | | | 1 | | |
| $l_5$ | | | 1 | 1 | | |
| $l_6$ | 1 | | | | 1 | |
| $l_7$ | | | 1 | | 1 | |
| $p(s_j)$ | 2 | 7 | 2 | 7 | 7 | 7 |

(b)

| | s | $S_N$ | $S_N$ | $S_N$ |
|---|---|---|---|---|
| $l_2$ | 1 | | | 1 |
| $l_5$ | 1 | 1 | | |
| $l_7$ | 1 | | 1 | |
| $p(s_j)$ | 2 | 5 | 5 | 5 |

(c)

**Fig. 2. Two-way edge reduction.**

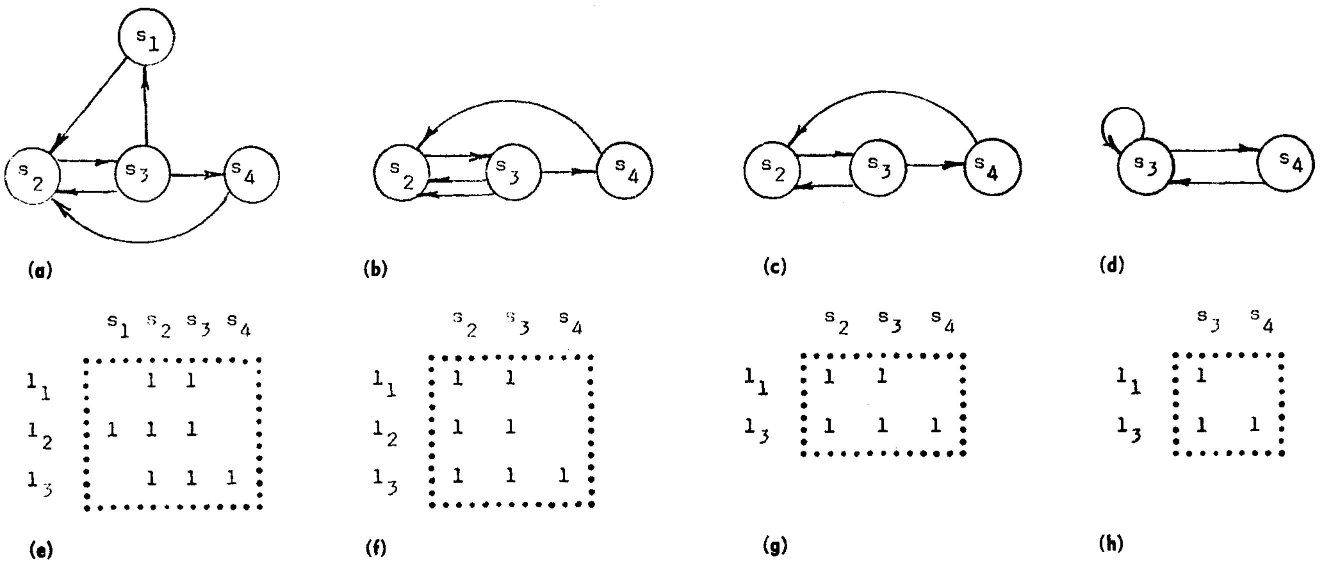



Fig. 3. Reduction of signal flow diagram.

## A BASIC TEARING ALGORITHM

In the previous section, simplifications to a loop matrix have been developed which can often bring a complete solution to a recycle problem. If the loop matrix to a process flow sheet graph is not available, the concept of row and column dominances may still prove useful. Their applications to a signal flow diagram is shown here as readily obtainable from the process flow sheet graph.

Given a process flow sheet graph G, its dual graph or signal flow diagram H is another finite directed graph in which the stream edges in G become the stream nodes in H. Two stream nodes $s_i$ and $s_j$ in $H$ are joined by a directed edge from $s_i$ to $s_j$ if and only if there exists a one-step path from the stream $s_i$ to $s_j$ in the graph G. Thus a set of stream edges which formed a simple loop in G will have the same set of stream nodes joined in a closed path by directed edges in the signal flow diagram. This preservation of simple loops under the transformation to the signal flow diagram means that a loop can be cut either in the process flow sheet graph or its signal flow diagram.

### Ineligible Stream Node

A stream node in a signal flow diagram is ineligible if it is dominated by another set of stream nodes, and a method by which it can be identified directly is given by the following theorem:

> "*Ineligibility Theorem*: If stream node $s_m$ has a weighting factor $p(s_m)$ which is larger or equal to the weighting sum of its immediate successors $T(s_m)$ or its immediate predecessors $T^{-1}(s_m)$, then it is ineligible."
>
> *Proof*: Consider a simple loop that passes through the stream node $s_m$. This loop must also pass through one node from its immediate predecessors $T^{-1}(s_m)$ and one node from its immediate successors $T(s_m)$. Therefore, the set of loops that pass through $s_m$ must also be contained in the set of loops that pass through $T(s_m)$ or $T^{-1}(s_m)$, that is,
>
> $$C_m \subseteq \sum_{i \in \Gamma} C_i$$
>
> is satisfied, where $\Gamma$ is the stream index set representing either $T(s_m)$ or $T^{-1}(s_m)$. Further if
>
> $$p(s_m) \geq \sum_{i \in \Gamma} p(s_i)$$
>
> then from (2), the stream $s_m$ is dominated by either $T(s_m)$ or $T^{-1}(s_m)$; therefore $s_m$ is ineligible.

Once a dominated stream is identified, its corresponding column in the loop matrix is deleted. Similarly a dominated row is also deleted. The extension of these reductions to a signal flow diagram can be shown to be equivalent to the following procedures:

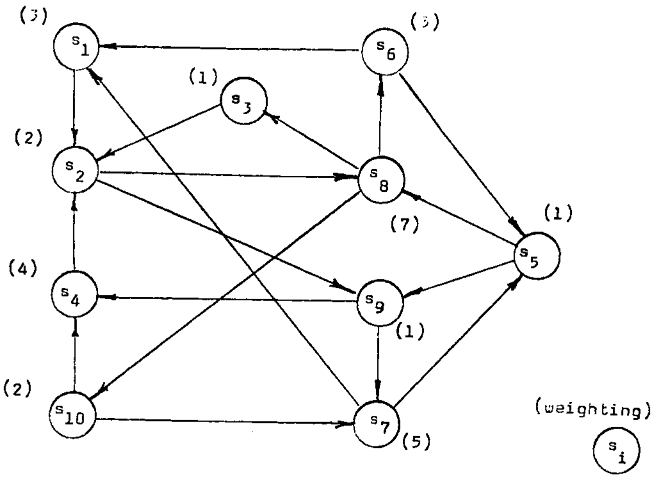
1. Delete the ineligible stream node $s_m$ from the signal flow diagram, and join the set of nodes in $T^{-1}(s_m)$ to the set of nodes in $T(s_m)$ by directed edges originating from $T^{-1}(s_m)$ and terminating on $T(s_m)$.
2. If more than two edges are directed from one stream node to another stream node, combine the edges into a single directed edge.

Figure 3a, for example, shows a signal flow diagram consisting of four stream nodes with all weightings set equal to unity. Figure 3e shows its loop matrix as consisting of three loops. Stream node $s_1$ is dominated by either $s_2$ or $s_3$, and it is removed by deleting $s_1$ and joining nodes $s_2$ and $s_3$ by a directed edge as shown in 3b. The equivalent deletion of $s_1$ from the loop matrix gives rise to the matrix in 3f. As loop $l_1$ is now dominated by $l_2$, it is also deleted which results in the matrix shown in 3g. This reduction is equivalent to the merging of two edges directing out from node $s_3$ and terminating on $s_2$ (Figure 3c).

### Essential Stream Node

One possibility which can result from the repeated removal of ineligible streams from a signal flow diagram is the appearance of nodes with a self-loop. Since a self-loop consists of only one stream node, it would appear in the loop matrix as a row with only one nonzero element at the corresponding stream column. Therefore any stream node with a self-loop must be essential and is a member of the optimal torn set. An essential stress stream column is removed from a loop matrix by deleting stream column $s_m$ and all the loop rows in $C_m$. Since the set $C_m$ represents all the loops that pass through $s_m$, an obvious equivalent reduction to a signal flow diagram is to delete node $s_m$ and all the edges connecting to it. This will automatically cut all loops that pass through node $s_m$.

In Figure 3c, the stream node $s_2$ is dominated by $s_3$, its




(a)

| | $s_1$ | $s_2$ | $s_3$ | $s_4$ | $s_5$ | $s_6$ | $s_7$ | $s_8$ | $s_9$ | $s_{10}$ | $p(s_i)$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $s_1$ | | 1 | | | | | | | | | 3 |
| $s_2$ | | | | | | | | 1 | 1 | | 2 |
| $s_3$ | | 1 | | | | | | | | | 1 |
| $s_4$ | | 1 | | | | | | | | | 4 |
| $s_5$ | | | | | | | | 1 | 1 | | 1 |
| $s_6$ | 1 | | | | 1 | | | | | | 3 |
| $s_7$ | 1 | | | | 1 | | | | | | 5 |
| $s_8$ | | | 1 | | | 1 | | | | 1 | 7 |
| $s_9$ | | | | 1 | | | 1 | | | | 1 |
| $s_{10}$ | | | | 1 | | | 1 | | | | 2 |

(b)

**Fig. 4. Signal flow diagram for Figure 1 and its adjacency matrix.**

removal results in the reduced graph shown in 3d which has a self-loop appearing on $s_3$. The equivalent loop matrix representation is shown in 3h. The removal of the essential stream $s_3$ from the signal flow diagram in 3d and from the loop matrix in 3h will cut all the loops. Therefore a solution to the recycle problem in Figure 3 is given by $s_2$ as the only torn stream.

### Basic Tearing Algorithm (BTA)

With the above preliminaries in hand, we are ready to state a Basic Tearing Algorithm (BTA) based only on the simplifications of the ineligible and essential stream nodes. This involves the steps:

(1) START: set graph $H$ equal to the signal flow diagram of the process flow sheet graph $G$.

(2) If $H$ is empty, STOP; otherwise untag all nodes in $H$ and go to step (3)

(3) Pick an untagged stream node $s_m$ from $H$; if all nodes are tagged, go to step (4).

(a) If stream $s_m$ is not ineligible, tag $s_m$ and return to step (3); otherwise delete $s_m$ from $H$ by means of the ineligible stream node reduction. Examine for nodes with a self-loop. If there is none, return to step (3); otherwise proceed to the next step.

(c) If $s_q$ has a self-loop, declare $s_q$ as a torn stream and delete $s_q$ from $H$ by means of the essential stream reduction.

(d) Remove from $H$ all nodes which have either no input or output edges. Remove also their connecting edges.

(b) Return to step (2).

(4) The graph $H$ is not reducible by the BTA.

### Algorithm Implementation

The BTA can be implemented on a computer in a number of different ways, depending on how a finite directed graph is stored in the computer (Evans et al., 1968). One method which has the advantage of simplicity and is suitable for hand calculation is based on the representation of the signal flow diagram by its adjacency matrix.

An adjacency matrix to a signal flow diagram is a zero-one matrix whose row $i$ and column $j$ represent the stream nodes $s_i$ and $s_j$, respectively. The $(i, j)$ element is set equal to 1 if and only if there is an edge directed from node $s_i$ to node $s_j$ in the signal flow diagram; otherwise it is set equal to zero. Therefore given a stream node $s_m$, its immediate successors are given by the columns which have nonzero elements in the row $m$; and the set of its immediate predecessors is given by the rows which have nonzero elements in the column $m$. A self-loop on a node $s_q$ is characterized by the appearance of 1 on the diagonal element $(q, q)$.

By the Ineligibility Theorem, we can determine the ineligibility of a stream node $s_m$ simply by scanning the row $m$ and column $m$ for its immediate successors and predecessors and comparing the weighting factors. If $s_m$ is dominated by its immediate successors $T(s_m)$, the ineligible stream $s_m$ is removed from an adjacency matrix by crossing out row $m$ and column $m$; then for each stream $s_p$ in $T(s_m)$, the elements in column $p$ are modified by forming a Boolean sum with the corresponding elements in column $m$. Similarly, if $s_m$ is dominated by its immediate predecessors $T^{-1}(s_m)$, it is removed from an adjacency matrix by crossing out row $m$ and column $m$, and for each $s_q$ in $T^{-1}(s_m)$, the elements in row $q$ are modified by forming a Boolean sum with the corresponding elements in row $m$. If $s_m$ is essential, it is removed from an adjacency matrix simply by crossing out row $m$ and column $m$.

To illustrate the BTA using the adjacency matrix, consider once again the recycle system shown in Figure 1. Its signal flow diagram and the equivalent adjacency matrix is shown in Figure 4. Both streams $s_1$ and $s_4$ are dominated by their immediate successor $s_2$. Removing both the ineligible streams results in the matrix shown in Figure 5a with rows 1, 4 and columns 1, 4 all crossed out. From the matrix in 5a, both streams $s_6$ and $s_7$ are dominated by their immediate successors $s_2$ and $s_5$. Removing $s_6$ and $s_7$ gives the matrix in 5b. Finally stream $s_8$ is observed from the matrix 5b to be dominated by its immediate successors $s_2$, $s_3$, $s_5$ and $s_{10}$. Removing $s_8$ gives rise to the matrix 5c. The last matrix 5c has two nonzero elements appearing on the diagonal entries (2, 2) and (5, 5) therefore both $s_2$ and $s_5$ have self-loops and belong to the optimal torn set. Removing the essential stream $s_2$ and $s_5$ by crossing out rows 2, 5 and columns 2, 5 will leave columns 3, 9, and 10 all without nonzero elements. This implies they are nodes without any input edges, and therefore they are removed by crossing out the respective rows and columns. This leaves an adjacency matrix with all rows and columns crossing out, implying an empty signal flow graph $H$. Thus the BTA terminates having identified $s_2$ and $s_5$ as the optimal torn set, the same result obtained earlier.

It is noted here that the step by step reduction of the BTA can be carried out all within the original adjacency matrix by merely ignoring the rows and columns which have been crossed out. This makes the algorithm ex-

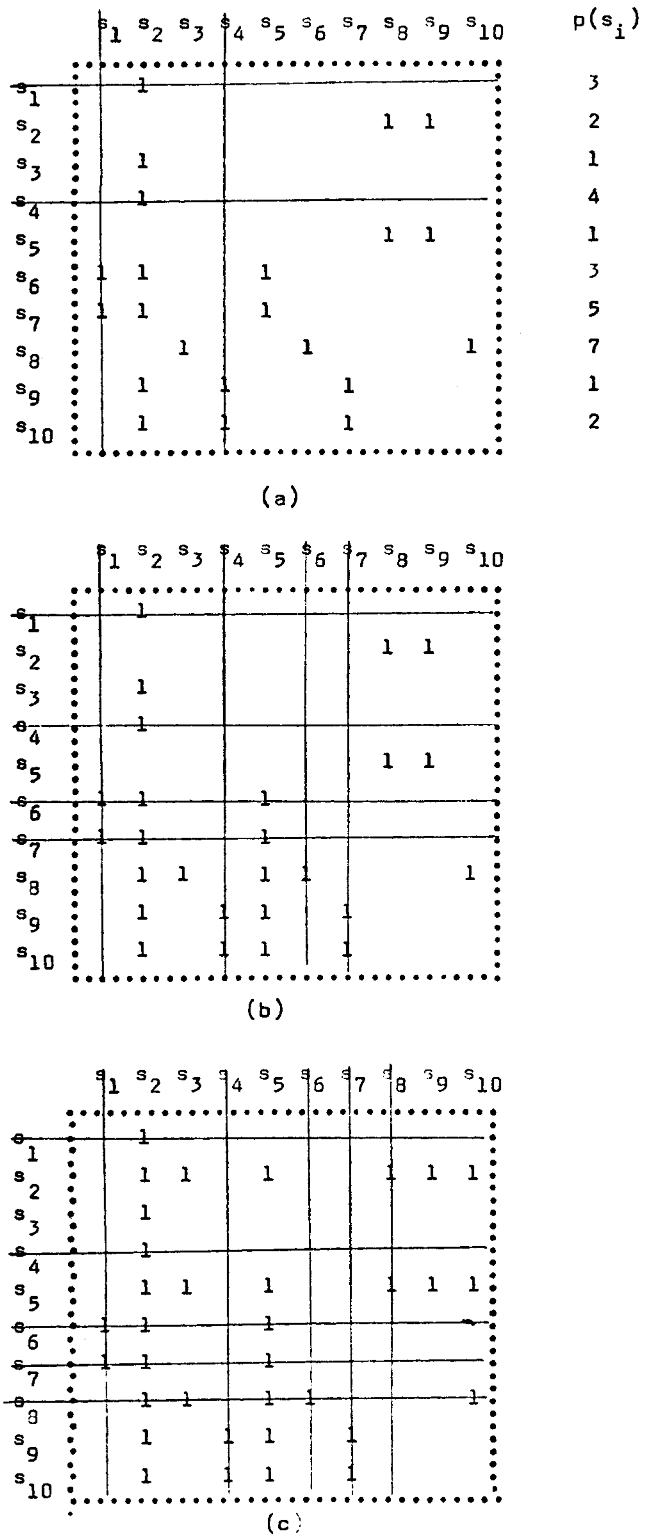



Fig. 5. Application of BTA using adjacency matrix.

tremely easy for hand calculation of moderately large recycle problems. Figures 6 through 8 are well-known examples taken from previous works (Sargent and Westerberg, 1964; Christensen and Rudd, 1969) that are successfully solved by the BTA. All the weightings used in these examples are assumed to be unity. The result is significant because the problems have been solved without additional reduction by the two-way edge required in previous methods.

## APPLICATION OF TWO-WAY EDGE REDUCTION

The Basic Tearing Algorithm fails when none of the stream nodes in the signal flow diagram can be identified as ineligible. It has been found, however, that in many cases considered the BTA can reduce the signal flow diagram sufficiently so that loops which are formed by two-way edges can appear. It was shown previously that only one stream from the two streams which form a simple two-way edge can be a recycle stream. For this reason we have chosen the stream $s_m$ with the lower weighting as a probable torn stream and removed $s_m$ from the signal flow diagram. At the same time the weighting of the

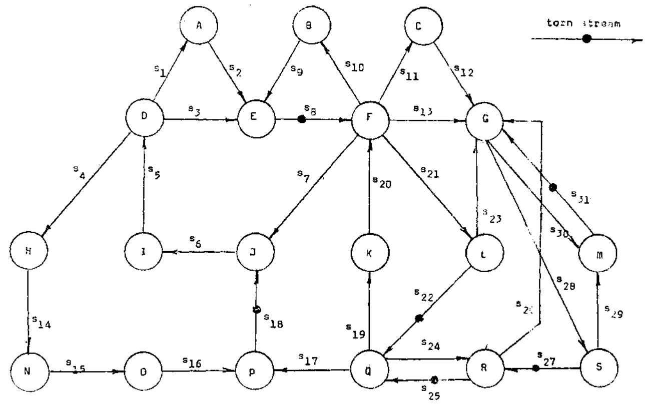



Fig. 6. Sargent and Westerberg example.

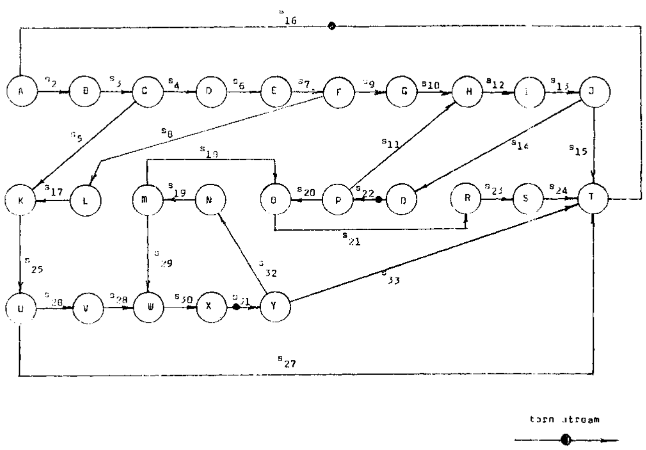



Fig. 7. Christensen and Rudd example No. 1.

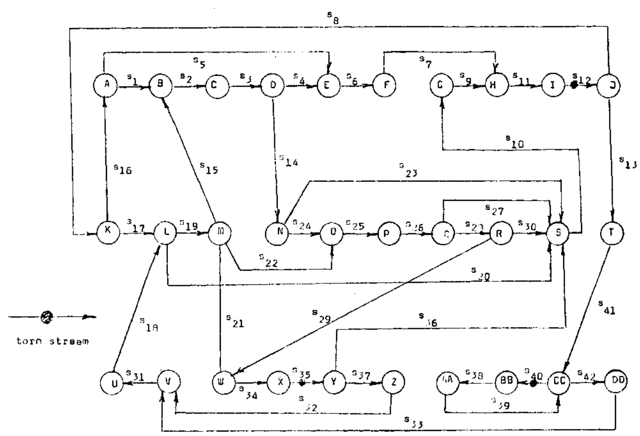



Fig. 8. Christensen and Rudd example No. 2.

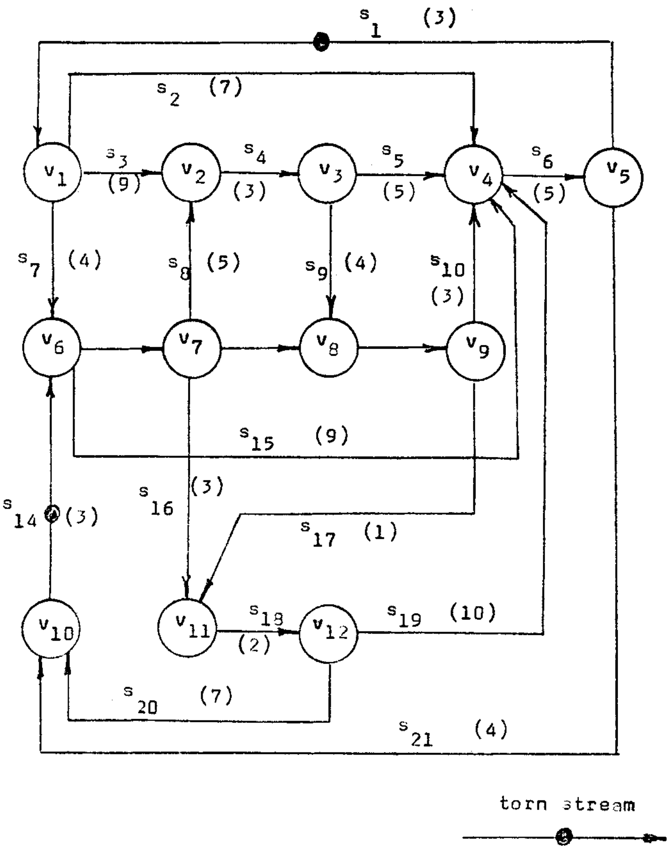



Fig. 9. Counter-example of Christensen and Rudd's algorithm.

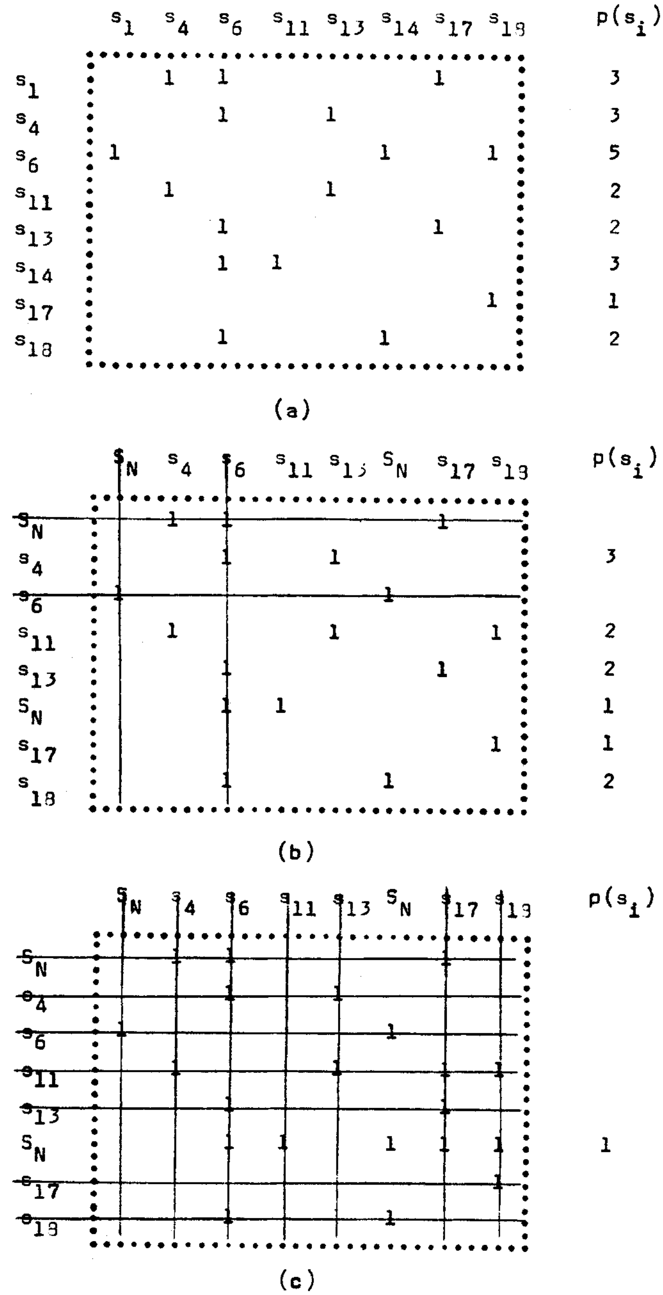



Fig. 10. An example for two-way edge reduction.

other stream $s_n$ which forms a simple two-way edge with $s_m$ is reduced by an amount $p(s_m)$. The BTA can then be applied to the resulting signal flow diagram. The probable torn stream $s_m$ becomes a true torn stream if its compliment stream $s_n$ is later found by the BTA as being ineligible; otherwise $s_m$ is discarded and $s_n$ becomes the torn stream.

If a two-way edge is not simple, then the modification as described previously must be followed. We shall illustrate the application of two-way edge reduction to the process flow sheet graph shown in Figure 9. The weightings in the graph are selected so that the system is not decomposable by the graph simplification technique of Sargent and Westerberg (1964) and Christensen and Rudd (1969).

A reduced signal flow diagram produced by applying the BTA is represented by its adjacency matrix shown in Figure 10a. No stream nodes can be found to be ineligible; therefore, the algorithm stops. Two pairs of two-way edge are formed between $s_1$ and $s_6$, and $s_{14}$ and $s_6$ with $s_6$ as the common node; therefore, they are not simple. We have $S_N = \{s_1, s_{14}\}$ and $p(S_N) = p(s_1) + p(s_{14}) = 6$ which is larger than $p(s_6) = 5$ where $s_6$ is the compliment node to $S_N$. Therefore we set $s_6$ as a probably torn stream and cross out row 3 and column 3 from the adjacency matrix. The column 1 is now without a nonzero element and is crossed out together with row 1. This gives rise to the matrix shown in Figure 10b. Application of the BTA will result in the final matrix shown in Figure 10c, which has a nonzero element appearing in the diagonal entry (6, 6). Since the column 6 has been named $S_N$, this implies the set of streams in $S_N$, that is, $s_1$ and $s_{14}$ are essential. This also means the probable torn stream $s_6$ which forms two-way edges with $S_N$ is not a torn stream. Removing the $S_N$ stream from the matrix in 10c as essential will make the signal flow diagram empty. Therefore, an optimal torn set to the recycle system in Figure 9 has been found to be $\{s_1, s_{14}\}$.

Another application of the BTA coupled with two-way edge reduction is the decomposition of the process flow sheet graph for a sulfuric acid plant in Figure 11 (Figure 7 in Barkley and Motard 1972). An optimal torn set when all the weightings are unity (minimum recycle streams problem) is found to be $\{s_4, s_{31}, s_{45}, s_{58}, s_{60}\}$. This set is obtained after one application of the two-way edge reduction by declaring $s_4$ as a probable torn stream. This stream is later confirmed as a true torn stream. By contrast, the five-member torn set found by Barkley and Motard is $\{s_{31}, s_{46}, s_{58}, s_{60}, s_{65}\}$. This demonstrates that the optimal torn set to a recycle problem is not necessarily unique. Sometimes an alternative torn set to the presently found set may prove to be more desirable. This can be achieved by the present method in a number of ways.

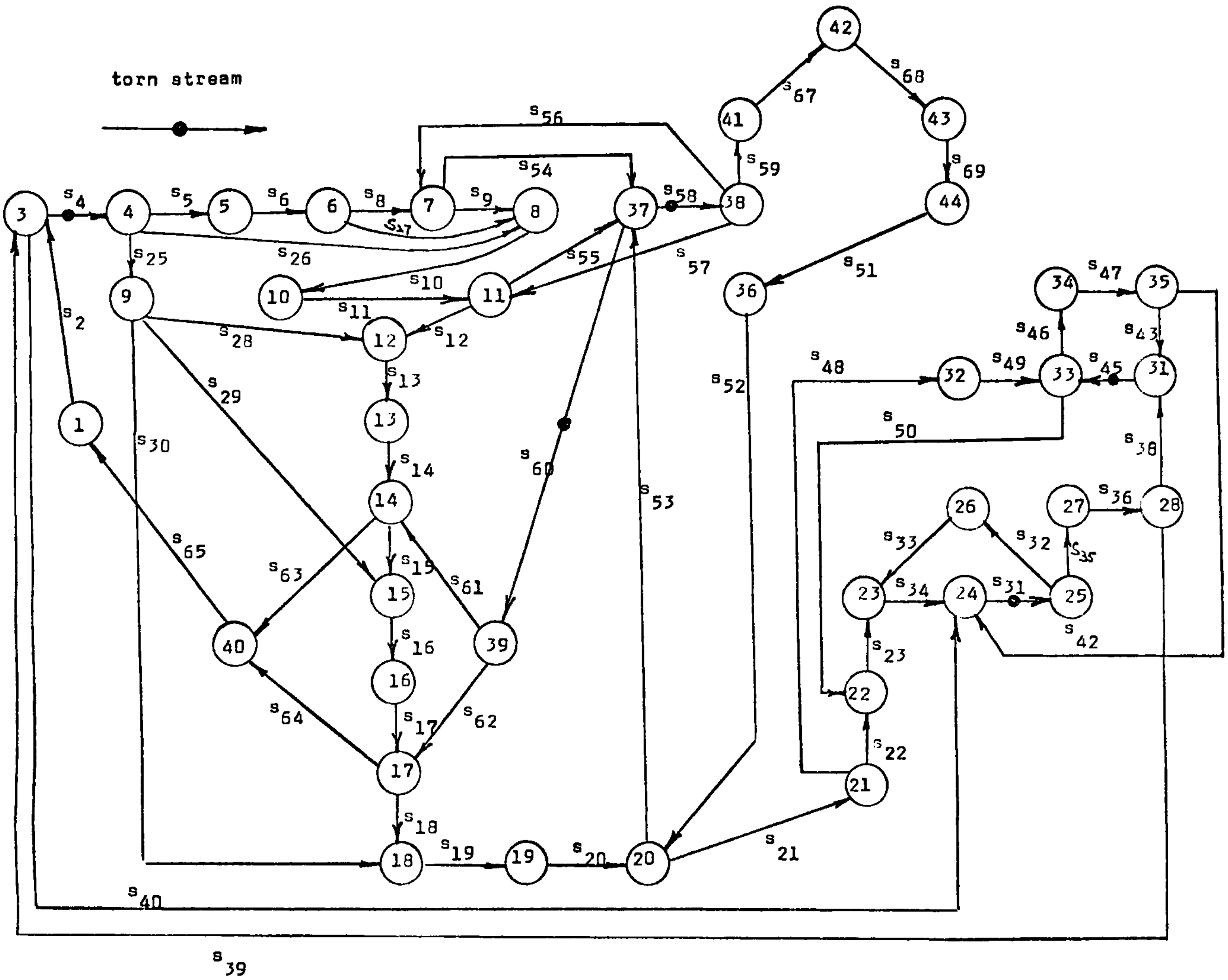



Fig. 11. Sulfuric acid plant example.

One is to declare, at the outset of applying the BTA algorithm, the unwanted torn stream as ineligible regardless of whether the stream satisfies the Ineligibility Theorem. Another is to assume an arbitrary high weighting to the unwanted torn stream. For example, if $s_{65}$ is the unwanted torn stream found from the Barkley and Motard's method, we can assign a weighting, say $p(s_{65}) = 3$, to $s_{65}$; the present method will then try to seek an alternative five-member torn set which excludes $s_{65}$. If this is not possible the algorithm will automatically generate a six-member torn set if there is one. We have shown here that a five-member torn set which excludes $s_{65}$ does exist for the sulfuric acid plant example. This flexibility of choosing an alternative optimal torn set is one useful feature offered by the present approach.

As a further illustration of the effectiveness of the BTA coupled with the two-way edge reduction, Table 1 lists an arbitrary set of weighting factors for the stream edges in the sulfuric acid plant example. For this set of weighting factors, it can be shown that the previous graph simplification techniques (Sargent and Westerberg, 1964; Christensen and Rudd, 1969) cannot reduce the process flow sheet graph completely. In addition, the problem size is too large for the dynamic programming algorithms (Sargent and Westerberg 1964; Upadhye and Grens, 1972) to be feasible. In the present method, application of the BTA alone identifies the following set of streams as belonging to the optimal torn set:

$$\{s_{32}, s_{38}, s_{40}, s_{43}, s_{58}\} \quad (3)$$

and generates a signal flow diagram which is represented by its adjacency matrix as

| | | | | | | | | | | | | $p(s_i)$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $s_5$ | | 1 | | | 1 | | | 1 | 1 | 1 | | 3 |
| $s_9$ | | | 1 | | | | | | | | | 1 |
| $s_{11}$ | | | | 1 | 1 | | | | 1 | 1 | | 2 |
| $s_{13}$ | | | | | 1 | | | | | 1 | | 1 |
| $s_{15}$ | | | | | | 1 | | | | | 1 | 2 |
| $s_{18}$ | | | | | 1 | | | | 1 | 1 | | 1 |
| $s_{25}$ | | | | 1 | 1 | 1 | | | 1 | 1 | 1 | 2 |
| $s_{29}$ | | | 1 | | | | | | | | | 1 |
| $s_{62}$ | | | | | | 1 | | | | | 1 | 2 |
| $s_{63}$ | 1 | | 1 | | | | 1 | | | | | 1 |
| $s_{64}$ | 1 | | 1 | | | | 1 | | | | | 3 |

(4)

In the matrix (4), the columns are named in the same order as its rows. The row 6 identifies stream $s_{18}$ as forming a two-way edge with streams in $S_{N_1} = \{s_{15}, s_{62}\}$, and row 10 identifies $s_{63}$ as forming a two-way edge with streams in $S_{N_2} = \{s_5, s_{11}, s_{25}\}$. Applying the two-way edge reduction to $s_{18}$ and $S_{N_1}$, and $s_{63}$ and $S_{N_2}$ will reduce the signal flow diagram to an empty graph with the stream set $\{s_{18}, s_{63}, s_{64}\}$ identified as an optimal torn set. Adjoining this set to the torn set in (3) will give an optimal torn set to the sulfuric acid plant example with weightings given by the values in Table 1.

TABLE 1. WEIGHTING FACTORS FOR SULFURIC ACID PLANT EXAMPLE

| $s_i$ | $p(s_i)$ | $s_i$ | $p(s_i)$ | $s_i$ | $p(s_i)$ |
|---|---|---|---|---|---|
| $s_2$ | 4 | $s_{25}$ | 2 | $s_{48}$ | 6 |
| $s_4$ | 5 | $s_{26}$ | 3 | $s_{49}$ | 1 |
| $s_5$ | 3 | $s_{27}$ | 1 | $s_{50}$ | 5 |
| $s_6$ | 5 | $s_{28}$ | 2 | $s_{51}$ | 7 |
| $s_8$ | 4 | $s_{29}$ | 6 | $s_{52}$ | 1 |
| $s_9$ | 1 | $s_{30}$ | 4 | $s_{53}$ | 4 |
| $s_{10}$ | 6 | $s_{31}$ | 7 | $s_{54}$ | 6 |
| $s_{11}$ | 2 | $s_{32}$ | 1 | $s_{55}$ | 5 |
| $s_{12}$ | 9 | $s_{33}$ | 3 | $s_{56}$ | 2 |
| $s_{13}$ | 1 | $s_{34}$ | 5 | $s_{57}$ | 3 |
| $s_{14}$ | 3 | $s_{35}$ | 3 | $s_{58}$ | 1 |
| $s_{15}$ | 2 | $s_{36}$ | 3 | $s_{59}$ | 2 |
| $s_{16}$ | 4 | $s_{38}$ | 2 | $s_{60}$ | 6 |
| $s_{17}$ | 6 | $s_{39}$ | 4 | $s_{61}$ | 4 |
| $s_{18}$ | 1 | $s_{40}$ | 1 | $s_{62}$ | 2 |
| $s_{19}$ | 5 | $s_{42}$ | 2 | $s_{63}$ | 1 |
| $s_{20}$ | 7 | $s_{43}$ | 1 | $s_{64}$ | 3 |
| $s_{21}$ | 3 | $s_{45}$ | 3 | $s_{65}$ | 5 |
| $s_{22}$ | 4 | $s_{46}$ | 3 | $s_{67}$ | 3 |
| $s_{23}$ | 6 | $s_{47}$ | 4 | $s_{68}$ | 5 |
| | | | | $s_{69}$ | 2 |

## BRANCH AND BOUND METHOD

In the event that the BTA coupled with two-way edge reduction fails to reduce a signal flow diagram, the final phase of the proposed algorithm calls for using a branch and bound method (Mitten, 1970). This method guarantees an optimal solution in all cases.

The branch and bound method is a clever enumerative scheme for solving optimization problems. Its advantage is derived from the fact that, in general, only a small fraction of the possible solutions need actually be enumerated, the remaining solutions being eliminated from consideration through the application of bounds that establish that such solutions cannot be optimal. Consider, for example, a problem which can be branched into two subproblems A and B, and whose solution includes the optimal solution to the original problem. If a lower bound $(\text{l.b.})_B$ to the problem B is readily obtained and it is known that the optimal solution to problem A is less than $(\text{l.b.})_B$, then it is clear that we need not solve for the optimal solution to problem B. This follows since the optimal solution to the original problem cannot be obtained by solving problem B.

The branching and bounding operation can be applied recursively to any subproblem whose optimal solution is not readily solved. The branching process will eventually lead to a subproblem which can be solved for its optimal value. Once this value is obtained, any subproblem whose lower bound is larger or equal to this value can be eliminated from further consideration. The subproblems whose lower bound is less than the optimal value must be further branched into their subproblems and have their new lower bounds determined. Two possible outcomes can result from the above procedure. One is that some subproblems are eliminated from further consideration and the other is that the remaining subproblems are solved for their optimal values. The optimal solution to the original problem is then given by the minimum of those subproblems solved.

### Lower Bound

The application of the branch and bound method to a recycle problem requires that a lower bound to an arbitrary signal flow diagram be evaluated efficiently. We can achieve this by observing that in order to solve for the variable in a stream $s_m$, all the stream nodes which immediately precede $s_m$ in the signal flow diagram must be specified. Therefore, if a signal flow diagram $H_R$ is not decomposable by the BTA, a good lower bound for decomposing $H_R$ is given by

$$(\text{l.b.})_{H_R} = \min_{\{s_i\}} \left[ \sum_{j \in \Gamma} p(s_j) \right] \tag{5}$$

where the minimization is taken over all the stream nodes in $H_R$ and $\Gamma$ is the stream index for the set of all the immediate predecessors to $s_j$, that is, $T^{-1}(s_j)$.

A signal flow diagram $H$ usually consists of a subgraph which is decomposable by the BTA and a remaining subgraph $H_R$ which is not. If a torn set is found by using the BTA, then its weighting sum must be added to the lower bound found by (5) to give a lower bound to the signal flow diagram $H$.

### A Worked Example

To illustrate the branch and bound method, we shall apply it to the reduced signal flow diagram after the BTA has been applied to the process flow sheet graph of the sulfuric acid plant shown in Figure 11. The adjacency matrix which represents this signal flow diagram is given in (4).

In order to define the branching operation, we observe that a stream node $s_m$ from a signal flow diagram must be either ineligible or essential. We can therefore define the following two subproblems:

Subproblem $G(s_m^{(i)})$:
All solutions to the original problem such that $s_m$ is ineligible.

Subproblem $G(s_m^{(e)})$:
All solutions to the original problem such that $s_m$ is essential.

It is obvious that the optimal solution must fall in one of the two subproblems. The stream node $s_m$ is chosen arbitrarily to be the node in the signal flow diagram with

| | | | | | | | | | | | $p(s_i)$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $s_5$ | | 1 | | | 1 | | 1 | 1 | 1 | | 3 |
| $s_9$ | | | 1 | | | | | | | | 1 |
| $s_{11}$ | | | | 1 | 1 | | | 1 | 1 | | 2 |
| $s_{13}$ | | | | | 1 | | | | 1 | | 1 |
| $s_{15}$ | | | | | | 1 | | | | 1 | 2 |
| $s_{18}$ | | | | | 1 | | | 1 | 1 | | 1 |
| $s_{29}$ | | | 1 | | | | | | | | 1 |
| $s_{62}$ | | | | | | 1 | | | | 1 | 2 |
| $s_{63}$ | 1 | | 1 | 1 | 1 | 1 | | 1 | 1 | 1 | 1 |
| $s_{64}$ | 1 | | 1 | 1 | 1 | 1 | | 1 | 1 | 1 | 3 |

(a) $s_{25}$ ineligible

| | | | | | | | | | | | $p(s_i)$ | $\delta(s_i)$ | $\sum_{i \in \Gamma} p(s_i)$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $s_5$ | | 1 | | | 1 | | 1 | 1 | 1 | | 3 | 7 | 4 |
| $s_9$ | | | 1 | | | | | | | | 1 | 2 | 3 |
| $s_{11}$ | | | | 1 | 1 | | | 1 | 1 | | 2 | 8 | 6 |
| $s_{13}$ | | | | | 1 | | | | 1 | | 1 | 3 | 2 |
| $s_{15}$ | | | | | | 1 | | | | 1 | 2 | 6 | 7 |
| $s_{18}$ | | | | | 1 | | | 1 | 1 | | 1 | 5 | 4 |
| $s_{29}$ | | | 1 | | | | | | | | 1 | 2 | 3 |
| $s_{62}$ | | | | | | 1 | | | | 1 | 2 | 5 | 6 |
| $s_{63}$ | 1 | | 1 | | | | | | | | 1 | 6 | 7 |
| $s_{64}$ | 1 | | 1 | | | | | | | | 3 | 4 | 4 |

(b) $s_{25}$ essential

Fig. 12. Example for the branch and bound method.

the maximum degree. From the adjacency matrix (4), stream nodes $s_{11}$, $s_{25}$, and $s_{63}$ all have the maximum degree of 8, from which we choose arbitrarily $s_m$ to be $s_{25}$. The adjacency matrices, after $s_{25}$ is assumed ineligible and essential, are shown in Figures 12a and 12b, respectively.

In the subproblem $G(s_{25}^{(i)})$, two self-loops appear on the stream nodes $s_{63}$ and $s_{64}$ after $s_{25}$ is taken as ineligible (Figure 12a). Application of the BTA after $s_{63}$ and $s_{64}$ have been removed will further identify $s_{18}$ as essential and reduce the adjacency matrix completely. Therefore the branching to the subproblem $G(s_{25}^{(i)})$ terminates with the optimal torn set found to be $\{s_{18}, s_{63}, s_{64}\}$ and the optimal value equal to 5.

In the subproblem $G(s_{25}^{(e)})$ where $s_{25}$ is assumed to be essential, the reduced adjacency matrix in Figure 12b is not decomposable by the BTA. The lower bound is computed using (5) and is found by taking the minimum value from the column where $\sum_{i\epsilon\Gamma} p(s_i)$ is computed (Figure 12b). This gives a value of 2. We add to this value $p(s_{25}) = 2$ because $s_{25}$ is taken as essential, thus giving a lower bound of 4 to the subproblem $G(s_{25}^{(e)})$. Since this lower bound is smaller than the optimal value of subproblem $G(s_{25}^{(i)})$, further branching of $G(s_{25}^{(e)})$ is required. From the column where the degree $\delta(s_i)$ is computed (Figure 12b), stream node $s_{11}$ has the maximum degree of 8. Therefore, we can branch $G(s_{25}^{(e)})$ into the following two subproblems:

Subproblem $G(s_{25}^{(e)}, s_{11}^{(i)})$:
The set of all solutions such that $s_{25}$ is essential and $s_{11}$ is ineligible.

Subproblem $G(s_{25}^{(e)}, s_{11}^{(e)})$:
The set of all solutions such that both $s_{25}$ and $s_{11}$ are essential.

It can be shown (Pho, 1973) that the lower bounds to the subproblems $G(s_{25}^{(e)}, s_{11}^{(i)})$ and $G(s_{25}^{(e)}, s_{11}^{(e)})$ are equal to 6 and 7, respectively. Because both are larger than the optimal value to subproblems $G(s_{25}^{(i)})$, we can eliminate all solutions to the subproblems $G(s_{25}^{(e)}, s_{11}^{(i)})$ and $G(s_{25}^{(e)}, s_{11}^{(e)})$. Therefore an optimal solution to the original problem where matrix (4) is the adjacency matrix is given by the solution to the subproblem $G(s_{25}^{(i)})$ with the optimal torn set as $\{s_{18}, s_{63}, s_{64}\}$ and the optimal value 5. This is the same solution as found by the two-way edge reduction described previously.

The solution steps involved in the branch and bound method are most conveniently represented by the decision tree shown in Figure 13. In this figure, the lower bound (l.b.) to each tree node as well as the torn set found to that stage is listed.

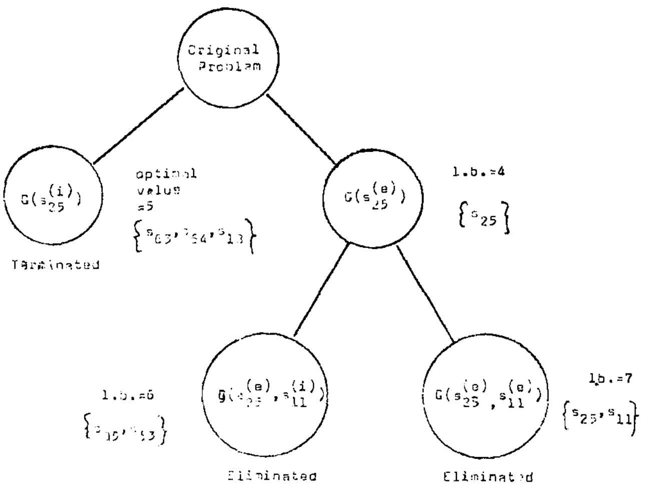



**Fig. 13. Decision tree for the branch and bound method.**

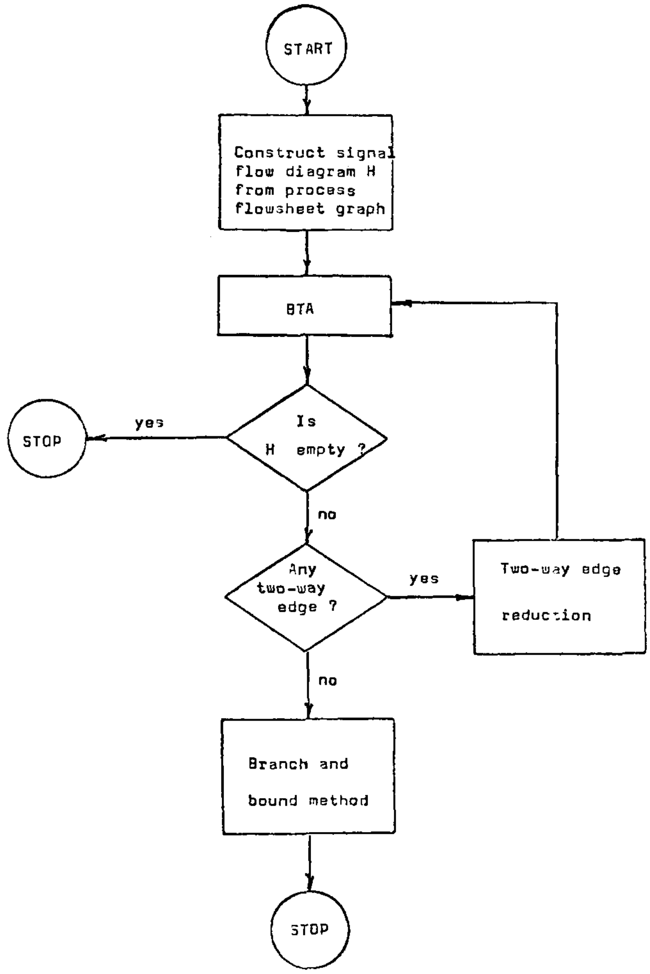



**Fig. 14. Flow chart for solving minimum recycle problem.**

## DISCUSSION

We have presented here certain systematic procedures, all based on the direct graph simplification of a signal flow diagram, to find a set of recycle streams which gives the minimum weighting sum. The BTA coupled with the two-way reduction is straightforward to apply and extremely efficient. Experience has shown that they are able to solve most of the cases considered (Pho, 1973). When the BTA fails, the branch and bound method will assure complete decomposition with few additional enumerations. We summarize the application of the proposed algorithm in the flow chart shown in Figure 14. Further specific details can be found in Pho (1973).

We have largely presented the algorithm in the form of a verbal description to emphasize that its actual implementation may vary in detail depending on the computer language used as well as the manner in which a finite directed graph is stored inside a computer. A method of implementing the BTA is described here using the adjacency matrix to represent the signal flow diagram. This enables the BTA and the two-way reduction to be performed entirely on a single matrix; therefore, it is suitable to a rapid manual decomposition of a moderate large size problem. All the examples in this paper were solved in this manner. No comparison in terms of computer time with the other tearing methods was attempted; however, several observations of their relative advantages can be made.

Since the exact analytical approach (Sargent and Westerberg, 1964; Upadhye and Grens, 1972) using dynamic programming to solve the minimum recycle problem suffers an inherent dimensionality difficulty which limits their use to problems of moderate size, we will not discuss them any further. The concept of the ineligible stream was first described by Christensen and Rudd (1969) who proposed the elimination of this ineligible stream directly on the process flow sheet graph where it is identified. A stream edge $e_{ij}$ is ineligible in a process flow sheet graph if (i) $e_{ij}$ is the only output stream of a node $v_i$ or the only input to a node $v_j$, and (ii) the weighting factor of stream $e_{ij}$ is greater or equal to the weighting sum of the input stream edges to $v_i$ or the weighting sum of the total output stream edges from $v_j$. Both $v_i$ and $v_j$ are two nodes in the process flow sheet graph. Transforming all the stream edges that are connected to nodes $v_i$ and $v_j$ into their equivalent signal flow diagram where the stream edges become the stream nodes, it can be shown readily by the Ineligibility Theorem that the stream $e_{ij}$ which is ineligible in the process flow sheet graph is also ineligible in the signal flow diagram. The converse, however, is not true as can be readily verified by considering nodes $v_i'$ and $v_j'$ when both have multi-input and output stream edges. The BTA which uses the Ineligibility Theorem as its means to identify ineligible streams is therefore a generalization on the concept of the ineligible stream. We must point out that the algorithm by Christensen and Rudd is performed directly on the process flow sheet graph and can be very efficient when it can reduce the graph completely. Otherwise the second part of their algorithm, which requires examining all the possible combinations of the index nodes, can involve severe combinatorial difficulty. We have also presented two examples which the first part of the Christensen and Rudd algorithm fails to resolve, whereas no difficulty is encountered by the present method.

When all the stream weightings are unity, the minimum recycle problem is reduced to finding a torn set which has the minimum number of streams. By the Ineligibility Theorem, any stream node in a signal flow diagram which has only one input or output edge is automatically ineligible. For a recycle system which has many process units with only one input or output stream, the BTA alone is often sufficient to decompose the signal flow diagram completely, as is evidenced from the examples solved in Figures 6 through 8. Similar decomposition techniques to find the minimum recycle streams were also proposed by Barkley and Motard (1972) using an interval reduction procedure. A set of stream nodes $S^+$ in a signal flow diagram will be reduced to an interval headed by a stream node $s_m$ if $s_m$ is the only immediate predecessor to each stream node in $S^+$. This interval reduction is identical to the reduction step we used to eliminate the stream set $S^+$ which is ineligible because each node in the $S^+$ has only one input edge. Using this concept of interval reduction, we note that if each stream node in a stream set $S^-$ has only node $s_n$ as its immediate successor, then we can reduce all the streams in $S^-$ into a new interval tailed by $s_n$. This generalization of the interval reduction is precisely the BTA when the latter is applied to a signal flow diagram with unit weightings. As a consequence of this generalization, the examples in Figures 6 through 8 have been solved without consideration of the two-way edge reduction, a step otherwise required by the Barkley and Motard algorithm.


## ACKNOWLEDGMENT

The authors wish to acknowledge the support of this work from the National Science Foundation Grant NSF-GK-24730. Furthermore, this work made use of the Princeton University Computer facilities supported in part by the National Science Foundation.


## NOTATION

$a_{ij}$ = $(i, j)$ element of loop matrix $A$
$A$ = loop matrix
$C_j$ = defined as $\{l_i | a_{ij} = 1\}$
$E$ = set of directed edges in G
$e_{ij}$ = directed edge from $v_i$ to $v_j$
$G$ = symbol for finite directed graph
$G(s_m^{(i)})$ = subproblem where $s_m$ is ineligible
$G(s_m^{(e)})$ = subproblem where $s_m$ is essential
$L$ = set of all simple loops
$l_i$ = $i$th simple loop in $L$
$M$ = number of loops in $L$
$N$ = number of streams in $G$
$p(s_j)$ = weighting of stream $s_j$
$R_i$ = defined as $\{s_j | a_{ij} = 1\}$
$s_j$ = $j$th stream
$S_N$ = $N$th stream set
$T(\bullet)$ = mapping for immediate successors
$T^{-1}(\bullet)$ = mapping for immediate predecessors
$V$ = set of nodes in G
$v_i$ = $i$th node in G
$x_j$ = optimization variable defined in (1)

### Greek Letters

$\delta(\bullet)$ = sum of $\delta^+(\bullet)$ and $\delta^-(\bullet)$
$\delta^+(\ )$ = in-degree
$\delta^-(\bullet)$ = out-degree
$\Gamma$ = stream index set

## LITERATURE CITED


Barkley, R. W., and R. L. Motard, "Decomposition of Nets," *Chem. Eng. J.*, **3**, 265 (1972).

Christensen, J. H., "The Structuring of Process Optimization," *AIChE J.*, **16**, 177 (1970).

———., and D. F. Rudd, "Structuring Design Computations," *ibid.*, **15**, 94 (1969).

Evans, L. B., D. G. Steward, and C. R. Sprague, "Computer-Aided Chemical Process Design," *Chem. Eng. Progr.*, **64**, No. 4, 39 (1968).

Ledet, W. P., and D. M. Himmelblau, "Decomposition Procedures for the Solution of Large Scale Systems," *Adv. Chem. Eng.*, **8**, 186 (1970).

Lee, W., and D. F. Rudd, "On the Ordering of Recycle Calculations," *AIChE J.*, **12**, 1184 (1967).

Mitten, L. G., "Branch and Bound Methods: General Formulation and Properties," *Oper. Res.*, **18**, 24 (1970).

Norman, R. L., "A Matrix Method for Location of Cycles of a Directed Graph," *AIChE J.*, **11**, 450 (1965).

Pho, T. K., "Topics in Computer-Aided Designs," Ph.D. dissertation, Princeton Univ., N. J. (1973).

Roth, R. H., "Computer Solutions to Minimum Cover Problems," *Oper. Res.*, **17**, 455 (1969).

Rubin, D. I., "Generalized Material Balance," *Chem. Eng. Progr. Symp. Ser., No. 00*, **58**, 43 (1962).

Rubin, J., "A Technique for the Solution of Massive Set Covering Problems with Application to Airline Crew Scheduling," IBM Tech. Report No. 320-3004 (1971).

Sargent, R. W. H., and A. W. Westerberg, "'SPEED-UP' in Chemical Engineering Design," *Trans. Inst. Chem. Engrs.*, **42**, 7190 (1964).

Upadhye, R. S., and E. A. Grens, II, "An Efficient Algorithm for Optimum Decomposition of Recycle Systems," *AIChE J.*, **18**, 465 (1972).

Weinblatt, H., "A New Search Algorithm for Finding the Simple Cycles of a Finite Directed Graph," *ACM J.*, **19**, 43 (1972).