τ_w = wall shear stress ($ML^{-1}T^{-2}$)

Literature Cited

(1) Heertzes, P. M.; Ven Mens, M. H.; Butaye, M. *Chem. Eng. Sci.* **1959**, *10*, 47.
(2) Clough, S. B.; Read, H. E.; Metzner, A. B.; Behn, V. C. *AIChE J.* **1962**, *8*, 346.
(3) Astarita, C. *Lincei-Rend. Sc. Fis. Inst. e nat.* **1964**, *36*, 361.
(4) Aiba, S.; Someya, J. *Makko Kogaku Zassi* **1965**, *43*, 603.
(5) Astarita, G. *Ind. Eng. Chem. Fundam.* **1965**, *4*, 236.
(6) Nishizima, Y.; Oster, G. *J. Polym. Sci.* **1965**, *19*, 337.
(7) Astarita, G. *Ind. Eng. Chem. Fundam.* **1966**, *5*, 14.
(8) Durill, P. L.; Griskey, R. G. *AIChE J.* **1966**, *12*, 1147.
(9) Aiba, S.; Someya, J. *J. Ferment. Technol.*, **1967**, *45*, 706.
(10) Quinn, J. A.; Blair, L. M. *Nature (London)* **1967**, *214*, 970.
(11) Srinivasan, N., M.S. Thesis, Illinois Institute of Technology, 1967.
(12) Hansford, G. S.; Litt, M. *Chem. Eng. Sci.* **1968**, *23*, 849.
(13) Luikov, A. V.; Shul'man, Z. P.; Puri, B. I. *Int. Chem. Eng.* **1968**, *14*, 493.
(14) Durill, P. L.; Grieskey, R. G. *AIChE J.* **1969**, *15*, 106.
(15) Luikov, A. V.; Shul'man, Z. P.; Puris, B. I. *Int. J. Heat Mass Transfer* **1969**, *12*, 377.
(16) Goldstick, T. K.; Fatt, I. *Chem. Eng Prog. Symp. Ser.* **1970**, *66*, 101.
(17) Hirose, T., Ph.D. Thesis, University of Waterloo, Canada, 1970.
(18) Zandi, I.; Turner, C. D. *Chem. Eng. Sci.* **1970**, *25*, 517.
(19) Dim, A.; Gardner, G. R.; Ponter, A. B.; Wood, T. *J. Chem. Eng. Jpn.* **1971**, *4*, 92.
(20) Navari, R. M.; Gainer, J. L.; Hall, K. R. *AIChE J.* **1971**, *17*, 1028.
(21) Grief, R.; Cornet, I.; Kapperser, R. *Int. J. Heat Mass Transfer* **1972**, *18*, 928.
(22) Grief, R.; Kapperser, R.; Cornet, I. *J. Electrochem. Soc.* **1972**, *119*, 717.
(23) Wasan, D. T.; Lynch, M. A.; Chad, K. J.; Srinivasan, N. *AIChE J.* **1972**, *18*, 928.
(24) Grief, R.; Paterson, J. A. *Phys. Fluids* **1973**, *16*, 1816.
(25) Paterson, J. A.; Grief, R.; Cornet, I. *Int. J. Heat Mass Transfer* **1973**, *16*, 1017.
(26) Parez, J. F.; Sandall, O. C. *AIChE J.* **1973**, *19*, 1073.
(27) Mashelkar, R. A.; Soylo, M. A. *Chem. Eng. Sci.* **1974**, *29*, 1089.
(28) Deo, P. V.; Vasudeo, K. *Chem. Eng. Sci.* **1977**, *32*, 328.
(29) Pathak, M. P.; Lal, P.; Upadhyay, S. N.; Mishra, P. Paper presented at the "Seminar on Transport Phenomena", Banaras Hindu University, Varanasi, India, July 28-30, 1978.
(30) Mishra, P.; Singh, P. C. *Chem. Eng. Sci.* **1978**, *33*, 1463.
(31) Vrentas, J. S.; Duda, J. L. *AIChE J.* **1979**, *25*, 1.
(32) Linton, W. H.; Sherwood, T. K. *Chem. Eng. Prog.* **1950**, *46*, 258.
(33) McAdams, W. H., "Heat Transmission", 3rd ed.; McGraw-Hill: New York, 1954.
(34) Wilkinson, W. L. "Non-Newtonian Fluids—Fluid Mechanics, Mixing, and Heat Transfer", Pergamon Press: London, 1960.
(35) Skelland, A. H. P. "Non-Newtonian Flow and Heat Transfer", Wiley: New York, 1967.
(36) Porter, J. E. *Trans. Inst. Chem. Eng.* **1971**, *49*, 1.
(37) Skelland, A. H. P. "Diffusional Mass Transfer", Wiley-Interscience: New York, 1974.
(38) Sherwood, T. K.; Pigford, R. L.; Wilke, C. R. "Mass Transfer", McGraw-Hill–Kogakusha, Ltd.: Tokyo, 1975.
(39) Pigford, R. L. *Chem. Eng. Prog. Symp. Ser.* **1955**, *17*, 79.
(40) Metzner, A. B.; Vaughn, A. P.; Houghton, G. L. *AIChE J.* **1957**, *3*, 92.
(41) Lyche, B. C.; Bird, R. B. *Chem. Eng. Sci.* **1956**, *8*, 35.
(42) Metzner, A. B.; Gluck, D. F. *Chem. Eng. Sci.* **1960**, *12*, 185.
(43) Christiansen, E. B.; Craig, S. E. *AIChE J.* **1962**, *8*, 154.
(44) Forrest, G.; Wilkinson, W. L. *Trans. Inst. Chem. Eng.* **1973**, *51*, 331.
(45) Forrest, G.; Wilkinson, W. L. *Int. J. Heat Mass Transfer* **1973**, *16*, 2377.
(46) Popovska, F.; Wilkinson, W. L. *Chem. Eng. Sci.* **1977**, *32*, 1155.
(47) Jost, W. "Diffusion in Solids, Liquids, and Gases", Academic Press: New York, 1969.
(48) Eubank, D. C., Proctor, W. S. S.M. Thesis in Chem. Eng., M.I.T., Cambridge, Mass., 1951.
(49) Burn, K. S., M.S. Thesis in Chem. Eng., University of Delaware, Newark, Del. 1960.
(50) Kumar, S., Ph.D. Thesis in Chem. Eng., Banaras Hindu University, Varanasi, India, 1976.
(51) Kumar, S.; Upadhyay, S. N.; Mathur, V. K. *J. Chem. Eng. Data*, **1978**, *23*, 139.
(52) Eisenberg, M.; Chang, P.; Tobias, C. W.; Wilke, C. R. *AIChE J.* **1955**, *1*, 558.
(53) Steele, L. R.; Geankoplis, C. J. *AIChE J.* **1959**, *5*, 198.
(54) Stephen, H.; Stephen, T. "Solubilities of Inorganic and Organic Compounds", Pergamon Press: New York, 1963.
(55) Kasaoka, S.; Nitta, C. K. *Kagaku Kogaku*, **1969**, *33*, 1231.
(56) Perry, J. H. "Chemical Engineer's Handbook", 4th ed.; McGraw-Hill: New York, 1963.
(57) King, C. V.; Brodie, S. S. *J. Am. Chem. Soc.* **1937**, *59*, 1375.
(58) Chang, S. Y., M.S. Thesis in Chem. Eng., M.I.T., Cambridge, Mass., **1949**.
(59) Wilke, C. R.; Chang, P., *AIChE J.* **1955**, *1*, 264.
(60) Venkateswaran, S. D.; Laddha, G. S. *Indian Chem. Eng.* **1966**, T33.
(61) Pradhan, A. A.; Heideger, W. J. *Can. J. Chem. Eng.* **1971**, *49*, 10.
(62) Lozar, J.; Laguerie, C.; Couderc, J. P. *Can. J. Chem. Eng.* **1975**, *53*, 200.
(63) Metzner, A. B. *Nature (London)* **1965**, *208*, 267.
(64) Li, S. U.; Gainer, J. L. *Ind. Eng. Chem. Fundam.* **1968**, *7*, 433.
(65) Hoshino, S. *Int. Chem. Eng.* **1971**, *11*, 353.
(66) Osmers, H. R.; Metzner, A. B. *Ind. Eng. Chem. Fundam.* **1972**, *11*, 161.
(67) Caskey, J. A.; Barlage, W. B. Paper presented at the 72nd National A.I.Ch.E. Meeting, St. Louis, May 1972.

Received for review February 14, 1979
Accepted October 12, 1979

New Algorithms for the Synthesis and Analysis of Fault Trees

David J. Allen* and M. S. Madhava Rao

Foster Wheeler Development Corporation, Livingston, New Jersey 07039

New and extended algorithms are proposed for the synthesis and analysis of fault trees. The use of these algorithms in computer programs facilitates fault tree analysis and allows the analyst to focus his attention upon the system's behavior. The algorithms have been applied to a major failure analysis performed upon a chemically active, fluidized bed coal/oil gasification unit. Trees containing 500 gates were synthesized and analyzed.

Fault Tree Analysis in the Chemical Industry

Public concern, government regulation, and the increasing complexity of chemical processing plants have led to an increased emphasis upon rigorous analysis as a means to identify the causes and likelihood of the occurrence of untoward events within a process, be they related to safety or to system availability. The use of logic or fault trees has been recognized as a powerful technique with which to perform such analyses (Lawley, 1974; Powers and Tomkins, 1974). However, outside the nuclear industry, the use of these techniques has been limited to a few groups, in large part because of the inability of widely used fault tree analysis computer programs to efficiently handle the detailed fault trees representative of the causes of failures in chemical plants and the difficulties of creating these fault trees. These difficulties are occasioned by the complexities introduced by the presence of control loops within a chemical process and are compounded by the fact that the mastery of algorithms for fault tree synthesis lies outside the usual expertise required of process design engineers.

In this paper we describe two computer programs and their algorithms that greatly facilitate the task of fault tree synthesis and analysis: first, an algorithm and program for fault tree synthesis that allows the analyst to concentrate upon the task of system definition; second, a fault

0019-7874/80/1019-0079$01.00/0

tree analysis program that is significantly faster than programs currently in use when applied to large, complex fault trees. These programs have been used to study a chemically active, fluidized bed gasification unit, synthesizing and analyzing fault trees containing up to 500 gates.

Fault Tree Synthesis

A fault tree is a graphical representation of the logical relationship between a specific event and its initiating or causal events. Through the analysis of the fault tree, the causes of the specific event can be determined as "minimal cut sets", i.e., as sets of events that are both necessary and sufficient for the specific event to occur.

The synthesis of fault trees is a deductive process: the logical combination of events that cause or precede the event of interest is determined and so, in turn, are the precursors of these events. The development of the tree is terminated when acceptable probability data are available for the causal event being examined or when this event may be regarded as being, in some sense, an initiating or primal event.

The manual synthesis of fault trees is, however, a complex and time consuming process that is prone to error particularly when the system contains many feedback and feedforward control loops or mechanisms. To facilitate fault tree synthesis, various algorithms have been devised. Some have been incorporated into computer programs, thus allowing the analyst to concentrate on system definition.

Among the algorithms proposed are those of Fussel (1973), Caceres and Henley (1976), Chu (1976), Powers and Tomkins (1974) Lapp and Powers (1977), and Camarda et al. (1978).

Fussell (1973) devised a synthesis technique for piecing together a fault tree from system independent component information beginning with the failure of interest and proceeding to more basic failures. In this technique the mini-fault trees or failure transfer functions that describe failure modes are synthesized into a single fault tree. This program has been successfully applied to simple electrical systems, but to apply it to complex chemical processes or energy processes is difficult because of problems that arise in the preparation of the failure transfer functions and in the necessary manual development of fault events.

Caceres and Henley (1976) created an algorithm by which fault trees can be constructed through the transformation of a block diagram. This algorithm is, however, unable to handle feedforward loops and, thus, is limited in its applicability to chemical processes or power plants. The fault tree construction techniques of Chu (1976) suffer from the same shortcoming. Other network analysis techniques have deficiencies that would restrict their use to simple systems [e.g., GERT (Nehem, 1973) cannot cope with redundant causes of system failure].

Lapp and Powers (1977) have described a program for fault tree synthesis for complex chemical processes. It considers the topology of the system to be analyzed, insofar as both feedback and feedforward control loops or mechanisms have been accounted for. Details of this program and their algorithm have not been made available but it would seem to have similarities to that presented here.

To extend the capability of fault tree synthesis programs, Powers and Tomkins (1974) suggested a method to construct fault trees using input–output models for equipment. However, this approach was not developed to the stage in which fault trees could be synthesized for complex chemical processes. The difficulty lies in the choice of a heuristic procedure that defines the impact of one variable or event upon another; fault tree synthesis entails symbolic as opposed to dynamic simulation of the process.

As has been noted, several of these algorithms have deficiencies that limit their applicability to chemical processes. However, two more general criticisms have been raised regarding fault tree synthesis algorithms. The first is that the synthesized fault-tree is unlike any that would be constructed by hand; even if correct, it is difficult to follow. The second criticism is that the task of preparation that precedes fault tree synthesis often equals the task of fault tree creation by hand. These criticisms, we believe, cannot be leveled at our algorithm.

An Algorithm for Fault Tree Synthesis

As did the algorithm proposed by Lapp and Powers (1977), this algorithm for fault tree synthesis utilizes a "digraph" representation of the system to be analyzed. A digraph (directed graph) is a set of nodes connected by directed arcs and thus is a representation of the system which is particularly convenient for computer processing. For chemical or energy producing processes, the nodes represent process variables, failures, and the system failure(s) or hazard(s) of interest. Relations between the nodes are embodied in the directed arcs between the nodes. These arcs may be conditional upon other events. The gain associated with each arc can be specified: if a positive deviation in one variable or the occurrence of the event represented by one node results in a positive deviation in a second variable or the occurrence of an event represented by a second node, then the gain along the arc between them is positive. Similarly, gains can be defined as being negative or zero: if a positive deviation in a variable or occurrence of an event results in a negative deviation in a second variable, then the gain is negative; if deviations in one variable or occurrence of an event have no direct effect upon a second variable, perhaps for certain conditions, then the gain between the nodes representing those variables or events is zero. In drawing a digraph, arcs with zero gains are omitted unless they are between nodes also connected by an arc with a nonzero gain and are conditional upon another event.

To simplify the task of digraph preparation and fault tree synthesis, only the direction of the gains will be considered: the gains are restricted to the values $+1$, 0, and -1. This is in contrast to the approach of Lapp and Powers (1977), who considered both the direction and magnitude of gains: in their algorithm, a gain of ± 10 was assigned to an arc representing the effect of an event that exceeds the ability of control loops to handle the event. Other arcs are restricted to the values of ± 1 or 0. This feature complicates the task of digraph preparation in the presence of multiple control loops if some but not all of these loops are able to handle the disturbance (e.g., additional cooling of a reactor may not suffice to control a runaway reaction but emergency venting may be adequate). In such instances it further requires that all the control loops that can act upon a given disturbance be identified prior to the final assignment of gain values. With our algorithm such complexities are avoided. The event whose occurrence exceeds the ability of the control action to handle is simply included in the listing of the circumstances under which the specific control action can fail. This simplification not only facilitates digraph preparation but also serves to emphasize that a fault tree is a symbolic as opposed to a dynamic simulation of the process. Engineering judgment will have to be exercised to define the behavior of the system and to decide whether or not events or changes in variables will be of such a magnitude or nature that other significant changes will arise because of their occurrence.

In creating a digraph, the following rules should be obeyed: Unless otherwise indicated, the normal state for the system pertains. In the digraph, nodes are identified with a single label. Where manual operation is required as part of a control or emergency shutdown procedure, this should be portrayed separately for each procedure: the operator is not represented by a single node.

With the completion of a digraph, a fault tree can then be synthesized. Although for the simple examples that are often presented in discussion of fault trees the use of digraphs may be unnecessary, for more complex systems a digraph allows the analyst to focus his attentions solely upon the task of system definition and in doing so present a coherent representation of the system in which errors are readily apparent.

The algorithm for fault tree synthesis from the digraph is comprised of a number of steps.

(1) The system, as represented by the digraph, is read in, analyzed and formed into a series of sequences of events by tracing the normal or conditional paths of events backwards from the event of interest. Essentially, this entails the application of Algorithm I of SPEED-UP (Sargent and Westerberg, 1964); a sequence of nodes terminates when it encounters a node with no inputs or a node that has been previously encountered in the same or another sequence. Limitations upon computer storage and the desirability of computational efficiency rule out the application of matrix methods (Norman, 1965; Himmelblau, 1966) to analyze the digraph and identify the loops.

(2) The negative feedforward and all feedback loops within the digraph are identified, together with the means by which the loop may fail or reverse its actions. These loops provide the control mechanisms that nullify or correct abnormal states in the system under study. Their failure allows the effects of equipment failures or other events to propagate through the system. Extended loops are created when positive feedback loops intersect negative feedback loops. This allows for adequate representation of control loops in the system.

(3) The fault tree is synthesized using the algorithm presented in Figure 1. The application of this algorithm commences with the node in the digraph that represents the event of interest. The paths to this node are traced, OR gates being created where there are multiple inputs to a node and AND gates where conditions are imposed upon arcs or when propagation of the effect of failures or abnormal events through the system requires the failure of control loops. In the algorithm for the synthesis of that portion of the fault tree representing the propagation of failures through negative feedback loops, each set of inputs to each node lying on the loops is handled separately in recognition of the fact that failures of the loop between the node of interest and the node at which the loop(s) is first encountered nullify the effect of the set of inputs as well as the loop. This necessary feature has also recently been emphasized by Lambert (1979).

In synthesizing the fault tree an additional feature is introduced in this program: it is required that the definition of nodes must be compatible with their appearance in the fault tree. For example, consider the digraph in Figure 2. If a fault tree were to be prepared for the event "pressure low", then this fault tree would be

pressure low
OR
temperature low rupture of vessel

However, for the event "pressure high", the fault tree

pressure high
OR
temperature high

must result, a "negative" vessel rupture is meaningless.

This situation is handled by identifying, in the input to the program, those nodes, representing both state variables and failures, that must be restricted in value. The value of each of these nodes is defined by the gain of the arc and the definition of the nodes. A value of ±1 is then assigned to the top event and values for each of the inputs to the gates within fault tree are calculated. Where these inputs represent nodes, this value is compared to the value, if any, assigned to the node. If it is incompatible, then that input to the fault tree is deleted.

This algorithm does not allow for the incorporation of Exclusive-OR gates in the tree to account for reversed action in loops. As Henley and Kumamato (1977) and Locks (1979) have pointed out, such gates introduce logical inconsistencies that can be resolved only through an appeal to physical explanations. Accordingly, it would seem appropriate to utilize physical explanations in order to eliminate Exclusive-OR gates entirely; as in most real systems the likelihood of the simultaneous occurrence of two inputs to an Exclusive-OR gate is much smaller than the likelihood of occurrence of a single input, the Exclusive-OR gate can be replaced by a simple OR gate. By eliminating Exclusive-OR gates we also avoid the "failed safe" conditions required by the algorithm of Lapp and Powers (1977). Utilizing the same physical explanations, we can similarly neglect the unlikely simultaneous occurrence of reversed action in several control devices in any one loop, a scenario included in Lambert's revision of the Lapp–Powers algorithm (Lambert, 1979).

The fault tree resulting from the application of this algorithm is edited so that no gates are extraneous. No gates have a single input or two or more identical inputs, and the number assigned to a gate reflects its level in the tree. To facilitate the piecing together of fault trees, the user can select the gate number at which the fault tree listing commences. The output format is compatible with that required for the following fault tree analysis program. It is also compatible with MOCUS (Fussell et al., 1974).

An Example of the Algorithm for Fault Tree Synthesis

To illustrate our algorithm for fault tree synthesis, we have applied it to the nitric acid cooler problem described by Lapp and Powers (1977), (Figure 3). A digraph representing the causes of a high nitric acid cooler exit temperature is presented in Figure 4. This digraph differs from that described by Lapp and Powers in several respects: first, only the direction of the gains and not their magnitude is considered; second, to illustrate the handling of multiple control loops, the digraph includes a representation of the response of the operator to abnormal temperatures; finally, additional detail on the effects of instrument malfunction is considered. It should be emphasized, however, that the digraph does not include all failures and events relevant to the system: e.g., such events as the erroneous setting of the temperature controller to a high valve and operation under manual control are omitted.

For this digraph, three negative feedback loops and one negative feedforward loop are identified. They are: (1) The automatic temperature control negative feedback loop (nodes 2–9–8–10–7–2) in which valve 5, controlling the flow of cooling water, opens in response to a high temperature.

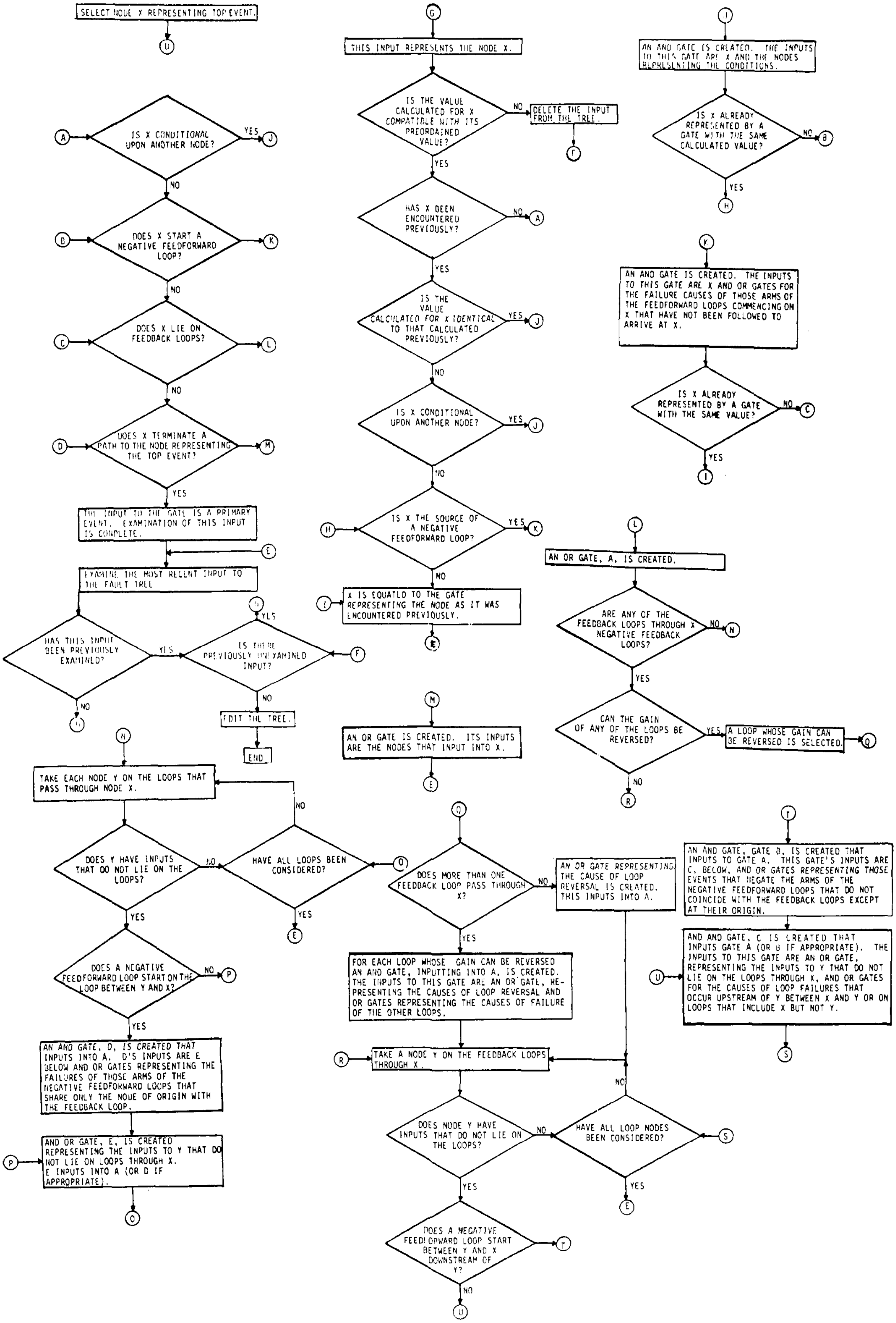

Figure 1. Algorithm for fault-tree synthesis.

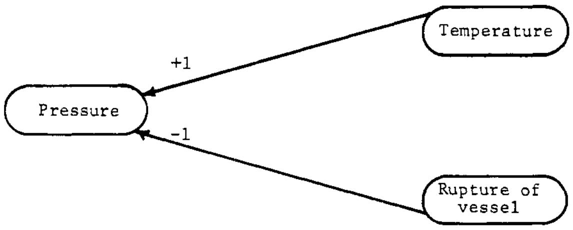

Figure 2. Simple digraph.

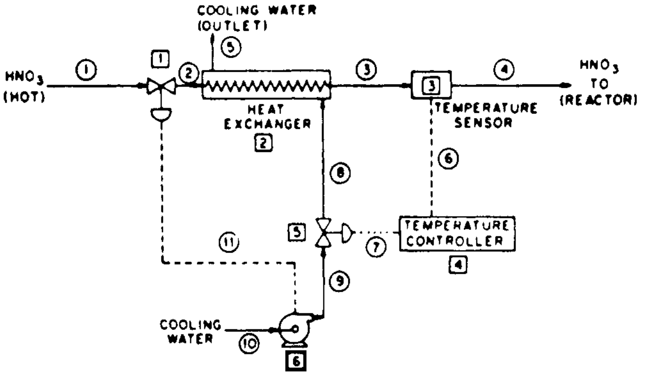

Figure 3. Nitric acid cooler.

(2) The negative feedback loop (nodes 2-9-8-6-7-2) in which the operator manually opens valve 5 in response to a high temperature. (3) The negative feedback loop (nodes 2-4-5-6-7-2) in which the operator reduces the flow of nitric acid when the cooler exit temperature is too high. (4) The negative feedforward control loop (nodes PE13-5-4-2 and PE13-9-2) that negates the effect of pump failure.

The fault tree shown in Figure 5 is then synthesized using the synthesis algorithm presented in Figure 1. Although this fault tree is small and thus relatively easy to create without the use of a fault tree synthesis program, we believe it provides a clear demonstration of the capabilities of our program.

Three limitations of this program (and of all other synthesis programs with which we are familiar) will be evident from the description of our algorithm and the above example: (1) Gate types other than AND or OR gates must be manually introduced into the tree at a later stage. (2) No distinction is made between various types of primary events. (3) Problems can occur in a quantitative analysis of the tree if component failures are repairable.

This last point can best be demonstrated using the example. Consider the event PE1, the failure of the temperature sensor such that a false low temperature is indicated. As this event alone is sufficient to cause the nitric acid exit temperature to rise, we need not be concerned with the duration of this failure. However, if we consider event PE2, the failure of the temperature indicator, we see it is of importance in two situations in which different probabilities must be assigned to the event. First, failure of the temperature indicator can act as a "trigger event" causing the nitric acid exit temperature to rise, should the operator respond to an apparently low temperature by increasing the flow of nitric acid to the cooler. In these circumstances, only the failure rate or occurrence frequency of the event is of interest. In a second situation, it is the unavailability of the temperature indicator with which we must be concerned: the nitric acid exit temperature will rise if the temperature controller fails calling for reduced water flow while the temperature indicator is unavailable.

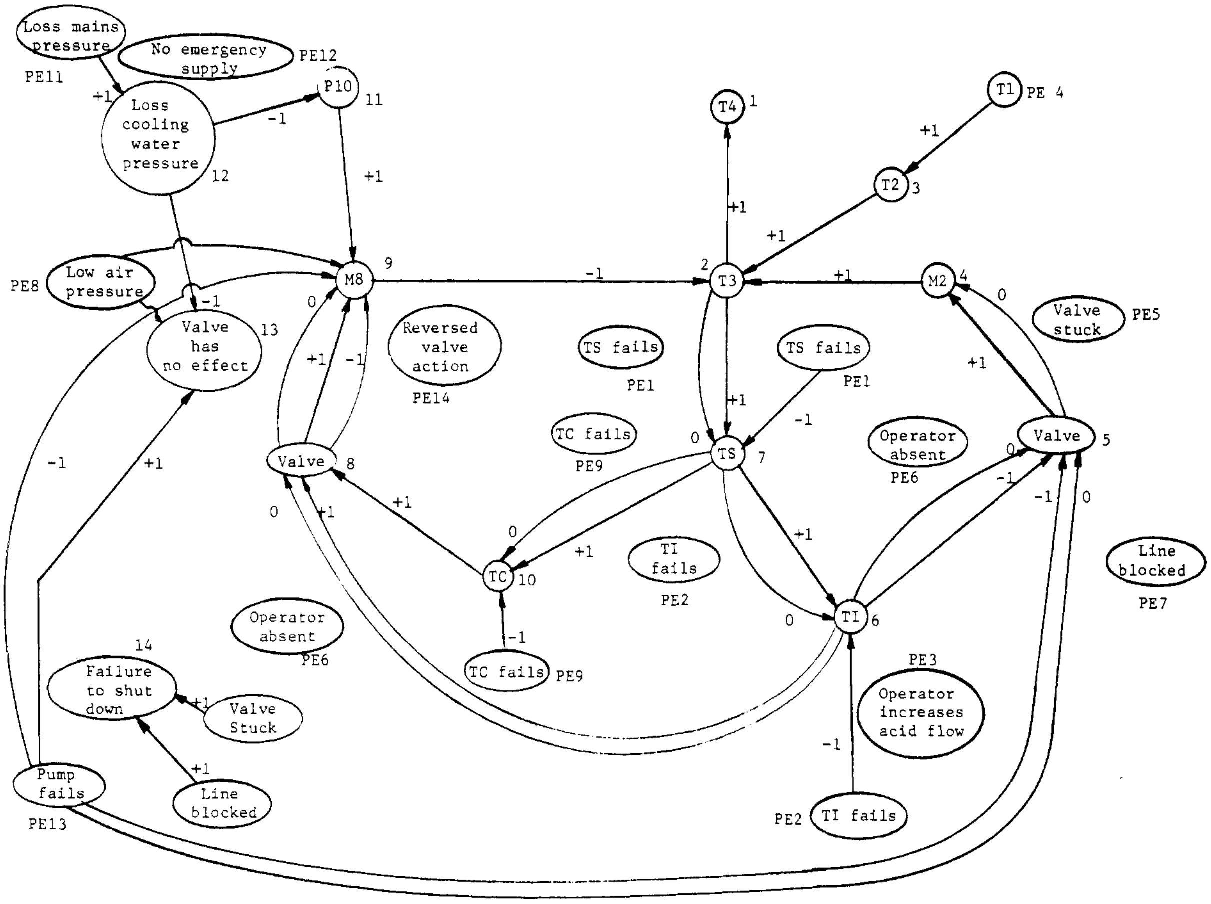

Figure 4. Digraph representing the nitric acid cooler process.

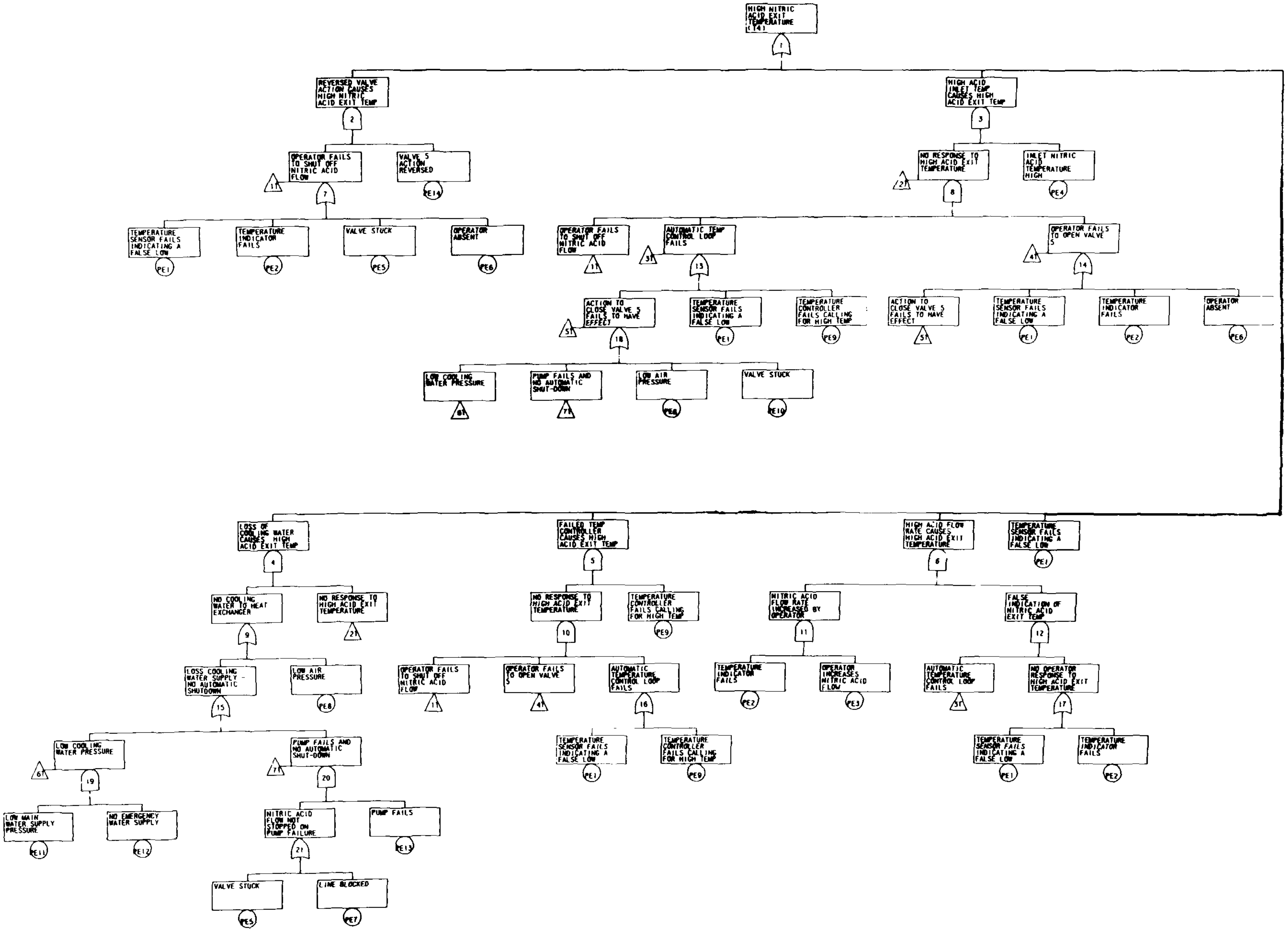

Figure 5. Fault-tree graphically representing the cause of a high nitric acid cooler exit temperature.

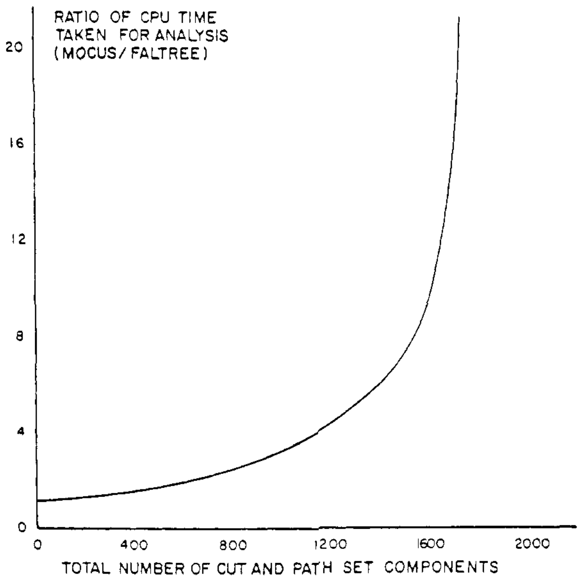

Figure 6. A comparison of MOCUS and our program (FALTREE).

Should this pose a problem, it is best remedied by modifying the tree following analysis and the identification of the causes (minimal cut-sets) of the particular system hazard or failure of interest. This task will, however, be facilitated if fault trees are synthesized using our algorithm as with it, each input to a control loop is examined individually.

Fault Tree Analysis

Gangadharan et al. (1977) and Wheeler et al. (1977) have described a computerized algebraic fault tree analysis technique that is significantly faster than the widely used program MOCUS (Fussel et al., 1974), particularly when analyzing larger fault trees (Figure 6). This technique is based upon the binary coding of events and bit manipulation for tree reduction. An upward moving algorithm for tree reduction is used. In this paper we describe an extension to these programs that further reduces the computation time for fault tree analysis and facilitates the examination of the fault tree.

This extension entails the identification of those primary events within the tree that are structurally equivalent to other primary events (e.g., primary events that always appear together as inputs to OR gates and do not appear elsewhere in the tree). The analysis is then performed, with the members of a set of equivalent primary events being replaced by a single representative primary event. This device not only speeds computation, halving the time required for analysis of test case fault trees, but it also allows the analyst to check if events really are structurally equivalent. Furthermore, it reduces the number of cut sets that need to be examined should the choice be made not to list those minimal cut sets that contain the remaining equivalent primary events, other than the representative event with which the analysis was performed.

In this manner we have obtained a fault tree analysis technique that is both fast enough to be applied without qualms to the complex fault trees that provide realistic representations of the causes of failures or other events in chemical processes and possesses features that aid the analyst in his examination of the tree.

An Application

The fault tree synthesis and analysis techniques described here have been used in the preparation and analysis of fault trees in the failure analysis of a chemically active, fluidized bed demonstration unit (Foster Wheeler Energy Corp., 1979). This analysis resulted in the synthesis of several fault trees, one containing 500 gates. Synthesis from the digraph of this largest tree and its subsequent analysis into minimal cut sets containing up to 4 elements required 30 s of CPU time on a CDC 6600 computer.

Although the time required for the synthesis and analysis of fault trees is both specific to the tree and highly dependent upon the complexity of the tree, we believe this figure to be indicative of the efficiency of these computer programs.

Literature Cited

Caceres, S., Henley, E. J., *Ind. Eng. Chem. Fundam.*, **15**, 128 (1976).
Camarda, P., Corsi, F., Trentadue, A., *IEEE Trans. Reliab.*, **27**, 215 (1978).
Chu, B. B., "A Computer-Oriented Approach to Fault Tree Construction", prepared for the Electric Power Research Institute, Report No. NP-288, EPRI Research Project 297-1, Palo Alto, Calif., 1976.
Foster Wheeler Energy Corporation, "Appendices 1 to 4 to Monthly Technical Progress Narrative No. 46", prepared for U.S. Environmental Protection Agency, EPA Contract No. 68-0202106, Report No. FWC/FWEC/R001-46, May 15, 1979.
Fussell, J. B., *Nucl. Sci. Eng.*, **52**, 421 (1973).
Fussell, J. B., Henry, E. B., Marshall, N. H., "MOCUS-A Computer Program to Obtain Minimal Sets from Fault Trees", ANCR-1156, Aerojet Nuclear Co., Idaho Falls, Idaho, 1974.
Gangadharan, A. C., Rao, M. S. M., Sundararajan, C., *Failure Prev. Reliab.*, 251 (1977).
Henley, E. J., Kumamoto, H., *IEEE Trans. Reliab.*, **R-26**, 316 (1977).
Himmelblau, D. M., *Chem. Eng. Sci.*, **21**, 425 (1966).
Lambert, H. E., *IEEE Trans. Reliab.*, **28**(1), 6 (1979).
Lapp, S. A., Powers, G. J., *IEEE Trans. Reliab.*, **26**, 2 (1977).
Lawley, H. G., *Chem. Eng. Prog.*, **70**(4), 45 (1974).
Locks, M. O., "The Lapp and Powers Fault Tree Model: S-Noncoherence", "Proceedings, Annual Reliability and Maintainability Symposium", IEEE, New York, N.Y., Jan 1979.
Nehem, R. F., "Gert-Graphical Evaluation and Review Technique, A Quantitative Hazard Analysis Tool", USAMC-ITC, Report No. 3-73-12, 1973.
Norman, R. L., *AIChE J.*, **11**, 450 (1965).
Powers, G. J., Tomkins, F. C., *AIChE J.*, **20**, 386 (1974).

Received for review April 20, 1979
Accepted August 17, 1979