NUREG/CR-1487
SAND80-1095
GM

VITAL AREA ANALYSIS USING SETS

Desmond W. Stack
Kimberly A. Francis

Date Published: May 1980

Sandia National Laboratories
Albuquerque, New Mexico 87185
operated by
Sandia Corporation
for the
U. S. Department of Energy

Prepared for
Division of Safeguards, Fuel Cycle and Environmental Research
Office of Nuclear Regulatory Research
U.S. Nuclear Regulatory Commission
Washington, D.C. 20555

NRC FIN No. A1060

8009030437

1-2

ABSTRACT

This report describes the use of the Set Equation Transformation System (SETS) for vital area analysis. Several concepts are introduced which enable the analyst to construct more efficient SETS user programs to perform vital area analysis. The advantages of performing the transformation of variables without first determining the minimal cut sets of the fault tree are discussed. A "bottom-up" approach to solving a fault tree is presented. The techniques described for vital area analysis are also suitable and efficient for many kinds of common cause analysis.

ACKNOWLEDGMENT

We want to thank R. B. Worrell, who designed and implemented the Set Equation Transformation System (SETS), for many helpful discussions and suggestions regarding these techniques.

CONTENTS

		Page
1.	Introduction	9
2.	Fault Tree Modelling and Analysis	10
3.	Vital Area Analysis	11
4.	Bottom Up Algorithm	13
5.	SETS Implementation	20
References		23-24

FIGURES

Figure | Page

1 Graphical Representation of Segment Relationships in a Fault Tree Plot 25

2 Significant Symbols of the Fault Tree Plot 26

VITAL A.῾A ANALYSIS USING SETS

1. Introduction

Vital area analysis [1, 2] is the analytical procedure used to systematically identify the areas of a nuclear power plant that require physical protection. As part of the U. S. Nuclear Regulatory Commission (USNRC) review of security plans, vital area analyses are being performed for all nuclear power plants. Sandia National Laboratories and Los Alamos National Scientific Laboratory are providing technical and computational support to the USNRC, Office of Nuclear Material Safety and Safeguards for the analysis. This report describes how to construct a Set Equation Transformation System (SETS) user program which can identify vital areas in a manner significantly more efficient than the previous SETS methodology.

Vital area analysis begins with a fault tree, F, whose primary events are sabotage actions that can lead to the undesired release of radioactive material. In addition to the fault tree, a set of Boolean equations, L, is required. Set L contains one equation for each primary event in F. This equation identifies the location, or logical combinations of locations, where that primary event can occur.

The increased efficiency of the SETS user program to be described is a result of the following conditions:

- The set L is added to the equation file after the fault tree is loaded but before the fault tree analysis is executed by a SETS user program.

- The fault tree analysis portion of the SETS user program implements a "bottom-up" solution rather than traditional "top-down" techniques.

2. Fault Tree Modelling and Analysis

The mathematical model of primary events and their combinatorial relationships is a set of interrelated Boolean equations. The fault tree is a graphical representation of this set. For the sabotage fault tree where primary events are sabotage actions, the set will be denoted by S.

The mathematical techniques used to analyze a fault tree model are applications of Boolean algebra, particularly the application of Boolean identities which produce equivalent Boolean equations. The use of SETS for fault tree analysis applies a minimal cut set algorithm [3] to the set S. The goal is to determine a Boolean equation which represents all of the minimal cut sets for the top event (or some other intermediate event) of the fault tree. Minimal cut sets for an intermediate or top event represent the fundamental ways that the event can occur in terms of the occurrence of primary events.

Since the analysis begins with a fault tree but is achieved using techniques based in Boolean algebra, some concepts and terminologies of fault tree analysis will be related to those of Boolean algebra.

The primary events of a fault tree are represented by independent Boolean variables in the set S. Each gate (intermediate event) in the fault tree is denoted by a dependent variable in the set S. Furthermore, each gate in the fault tree corresponds to an equation in S; the left side is the gate's dependent variable, and the right side is a logical combination of the variables representing the gate's inputs. The logical combination of variables is determined by the gate symbol (OR, AND, XOR, etc.) which is seen on the fault tree plot.

By the process of repeated substitution, any dependent variable can ultimately be expressed as a logical combination of independent Boolean variables. This is equivalent to saying that any intermediate event can be expressed as a logical combination of primary events. After the equation for a dependent variable is determined solely in terms of independent variables, a sequence of equation manipulations (called expansion and simplification, in [3]) produces a Boolean equation in disjunctive normal form. This equation is tantamount to a listing of the minimal cut sets for the gate represented by the dependent variable. The terms of the equation consist of only independent variables interpreted as primary events.

3. Vital Area Analysis

The previous methodology for vital area analysis sought first to derive a Boolean equation for the top event in terms of the primary events (sabotage actions) from the original fault tree. However, because of the size and complexity of most fault trees, this method requires excessive amounts of computer and analyst time. Furthermore, it is entirely possible that the Boolean equation which represents all minimal cut sets may be unobtainable despite the advanced techniques described in Section 4 of [3]. When this occurs, truncation must be employed, meaning that terms of order "n" or more* are ignored. In short, the Boolean equation which represents the solution to the fault tree is not complete. Such a truncated solution would make the subsequent vital area analysis incomplete also, that is, some of the vital locations may not be identified.

The new methodology for vital area analysis does not attempt to solve the Boolean equations derived from F. Instead, the set $S \cup L$ of equations is solved. (S is the set of equations derived from F.) This amounts to solving a location fault tree, i.e., a fault tree with locations or areas as primary events. The set $S \cup L$ defines such a fault tree, because each primary event (independent variable) in S is further developed into areas by an associated equation in L. Thus, the variables which are independent in S become dependent variables in $S \cup L$ and locations are represented by the independent variables of $S \cup L$.

In contrast to the location tree in Figure 3.6 of [1], the location fault tree defined by $S \cup L$ is larger than

*n is a positive integer chosen by the analyst.

the original fault tree. The difference is due to the fact that the location tree allows primary events of the original fault tree to be replaced by locations or logical combinations of locations. The location tree includes no sabotage actions, only those locations of possible action. Once this replacement is made, the original fault tree structure is changed--gates are coalesced or removed entirely depending on inputs. While this step reduces the general complexity of the fault tree, it also requires the formation of the location tree.

The location fault tree, set $S \cup L$, uses the primary events of the original tree and further develops each primary event into logical combinations of locations where they can occur. Although $S \cup L$ defines a larger fault tree, it is much easier to solve (i.e., find the Boolean equation representing all minimal cut sets of the top gate) since there are substantially fewer primary events. A typical reactor sabotage fault tree has over 200 primary events which represent sabotage actions, but only about 30 areas for these actions to occur. The "bottom-up" solution technique exploits this situation by developing every equation only in terms of independent variables. Since there are approximately only 30 independent variables, even large equations simplify to a manageable size at each step of the procedure. Finally, it is not necessary to plot the fault tree defined by the set $S \cup L$, since the "bottom-up" technique uses only the information available in the plot of the original fault tree.

4. Bottom Up Algorithm

In general, the "bottom-up" solution technique is suitable for a fault tree with a large number of replicated primary events relative to the tree's total number of primary events. An approximate ratio of the number of replicated primary events to the total number of primary events, used as a guide post for applying the bottom-up technique, has not been established; but the fault tree defined by S ∪ L has nearly all its primary events replicated, making it an excellent candidate for this approach.

The "bottom-up" algorithm seeks to determine an ordering on attempted gate solutions which guarantees that each gate solution equation consists of only primary events (independent variables). Vital area analysis using this technique begins with a careful examination of the (original sabotage) fault tree plot. In order to clarify the procedure, the following definitions are necessary.

The <u>fault tree plot</u> is comprised of several fault tree segments. (See Figure 1 for a graphical representation of segment relationships within a plot.) The definitions of the different types of fault tree segments are based on the plot created by the <u>Fault Tree Drawing Program</u> (FTDP) [4].

A <u>fault tree segment</u> is a connected set of gates for which one gate (the top) is developed until its branches terminate in primary events or other events, defined by user criteria to be treated as primary events.

There are three distinct types of segments for vital area analysis fault trees. The Major Fault Tree Segment (MFTS) develops the top gate of the fault tree down to primary events and transfer-in symbols. A transfer-in symbol made by the FTDP denotes the input of a replicated gate. An Intermediate Fault Tree Segment (IFTS) has a replicated gate as its top, with an associated transfer-out symbol. This multiple output event is developed down to primary events and transfer-in symbols. An IFTS contains at least one transfer-in symbol. A Terminal Fault Tree Segment (TFTS) also has a replicated gate as its top, with an associated transfer-out, but is developed down to only primary events.

In fault trees with a single top event and at least one replicated event, there is only one MFTS, and there is at least one TFTS. In fault trees with a single top event and no replicated events, the MFTS is the same as the TFTS. It is assumed that vital area analysis fault trees have at least one replicated gate, yielding a distinct MFTS and TFTS. Every replicated gate defines the top of a fault tree segment, consequently the fault tree decomposition is unique.

The MFTS contains the top event of the fault tree and is drawn by the FTDP as the leftmost segment on the plot. Each transfer-in located within the MFTS corresponds to a transfer-out symbol associated with the top gate of an IFTS or a TFTS. The transfer-out symbol is located at the top of the IFTS or TFTS and its gate is plotted down to other transfer-ins or primary events. Each segment is plotted to the right of the previous one, but in no particular order. TFTS's are often the rightmost segments of the plot.

It is assumed that the Forced Transfer Option of the FTDP is not used when plotting the fault tree. This option enables gates with a single output to be plotted with a transfer-out symbol. If this option is not used, all and only those gates with multiple outputs are plotted with a transfer-out symbol. Also, it is visually helpful if the Levelling Option of the FTDP is not used. This option causes segments to be drawn where they fit on the plot page, other than at the top of the plot. Occasionally this option will reduce the amount of paper used in a plot, but it forces the analyst to search the entire plot for occurrences of IFTS's and TFTS's. Without the option, the top symbol of each segment (major, intermediate, terminal) is plotted on the top edge of the plot sheet so the identification of every fault tree segment, no matter which type, is straight-forward.

Once the original fault tree (sabotage) is plotted, the next step of the analysis is to algorithmically determine the order in which gate solutions are to be attempted. The "bottom-up" algorithm concentrates on providing gate solution equations whose right-hand side consists of only independent Boolean variables. This is accomplished by beginning with the TFTS's at the bottom of the fault tree and building the solution, segment by segment, to the top gate.

The fault tree solution being sought is a representative Boolean equation for the top gate which is a minimal Boolean expression of independent variables in disjunctive normal form. A segment solution is a similar concept except the equation is for the top gate of the segment. The equation represents the family of all minimal cut sets, the fundamental ways the event can occur.

The methodology for finding a "bottom-up" fault tree solution is based on the identification and solution of a certain sequence of all of the fault tree segments. Each event of the fault tree is contained in at least one segment and the set of all fault tree segments completely cover the tree.

Viewing the fault tree as a set of segments, each segment can be labeled "solved" (subset D) or "unsolved" (subset N). Initially all segments are unsolved, i.e., $D = \Phi$. Ultimately, to obtain the fault tree solution, all segments are members of D, but to be efficient, the order of attempted solution is important.

The essential criteria for determining this sequence is that no segment solution be attempted unless it contains only primary events or all of its replicated gates (transfer-ins) have been previously solved.

The TFTS's are replicated gates developed solely into primary events. For this reason, the TFTS's are the first to be solved. They may be attempted in any order, using the techniques shown in [3]. These TFTS's become members of subset D, the "solved" subset. The pool of segment solutions is used to continue the analysis process by providing transfer-in solutions for other segments.

The analyst determines the next segment to solve based on the segment's transfer-in symbols and the subset of solved segments. All TFTS's are solved, so the next attempted solution is an IFTS with only TFTS's as transfer-ins. Once solved, the IFTS is a member of D and is available as a transfer-in solution to any other IFTS or the MFTS.

The solution process continues, decreasing the unsolved subset, N, until one segment is left, the MFTS. The technique to solve this segment (which is the fault tree solution) requires identification and solution of several intermediate events. The procedure is necessary partly because of the size and complexity of the MFTS, but is dependent upon the number and complexity of the transfer-in solutions. Solving the MFTS in this fashion reduces the number of terms in the solution equation at each step.

Again, this portion of the solution process requires examination of the fault tree plot. It is necessary to identify the intermediate gates to be solved and determine the solution sequence before attempting a top gate solution. The following algorithm successfully identifies all such gates.

Begin by finding all transfer-in symbols in the MFTS. Follow each symbol's output upward until an AND gate (see Figure 2) is encountered (possibly going through several gates of other types). Form a list containing all AND gates encountered in this manner, but do not duplicate gates. This list identifies the intermediate gates which are to be solved; the sequence of solution is determined by the transfer-in symbol's distance from the top gate.

Starting at the lowest level of the MFTS (the top gate is the highest), examine the segment horizontally. The FTDP plots equi-level* events on the same horizontal line. If this horizontal line contains any gates of the list, solve them (in any order). Continue moving toward the top gate (level-by-level), solving identified gates at each level that contains them. Once the list is exhausted, the top gate is solved. The resultant equation is the fault tree solution.

An outline for application of the "bottom-up" fault tree solution technique is as follows:

A. Obtain a fault tree plot to be analyzed.

B. Identify all TFTS, put them at the top of a solution list. This list will ultimately become the solution sequence for SETS implementation.

C. Identify an IFTS which has all its transfer-ins on the solution list. Put this IFTS on the solution list.

D. Repeat C until all IFTS's are on the solution list.

E. Identify all intermediate AND gates in the MFTS which are eventual outputs of its transfer-ins.

---

*One can define the concept of "level" of an event as a distance, i.e., the minimum number of gates plus one encountered from the output of an event to its eventual input into the top gate. The top gate is at the zero'th level.

F. Add these gates to the solution list based on their distance from the top gate--those farthest away are added first, those nearest are added last.

5. SETS Implementation

In vital area analysis, each sabotage action (primary event) is developed into a location or logical combination of locations where the action could occur. This, in effect, transforms the primary events of the fault tree from sabotage actions to sabotage action locations. Sabotage actions become intermediate events (gates) of the fault tree. Event development is accomplished by loading an additional block of equations which contains a transformation equation for each primary sabotage event.

To implement the bottom-up technique, the first lines of a SETS user program read and load the fault tree as well as a transformation block:

```
RDFT(SABOTAGE-FT).
RDINPBLK(LOC-TRANS).
LDBLK(LOC-TRANS).
```

The pattern for the solution of all gates (tops of segments as well as intermediate gates) is:

```
SUBINEQN(GT,GT).
REDUCEQN(GT,GT).
```

It is important to notice that the new equation (second parameter) replaces the old one. This insures the correct intermediate equation is propagated toward the solution.

Following the order established by the algorithm (see Section IV), the SETS user program consists of a SUBINEQN and REDUCEQN for each gate identified by the algorithm. The general outline for the program is:

A. Load the fault tree, load the transformation block;

B. SUBINEQN and REDUCEQN on all TFTS top gates (any order);

C. SUBINEQN and REDUCEQN on all IFTS top gates in the correct order (Section IV);

D. SUBINEQN and REDUCEQN on specified gates in the MFTS in the correct order;

E. SUBINEQN and REDUCEQN on the top fault tree gate;

F. Form a block containing the solution equation.

G. Print the solution equation.

The use of the techniques described in this report has resulted in considerable savings in both analyst time and computer time for the vital area analyses conducted thus far. Computer run times have been reduced from over 1,000 seconds to less than 200 seconds for the majority of the location fault trees analyzed using this approach. Similarly, analyst time required to solve a typical location fault tree has been reduced from several days to several hours. Before the techniques described in this report were employed, the minimal cut set equation for the sabotage fault tree often had to be truncated before a set of vital areas could be identified. The use of truncation implies that some of the vital areas may not be included in the set of vital areas identified. The location fault trees analyzed using the techniques described in this report have all been solved without resorting to truncation.

References

1. Varnado, G. B. et al., Reactor Safeguards System Assessment and Design, Volume I, June 1978, SAND77-0644, Sandia Laboratories, Albuquerque, NM.

2. Varnado, G. B., and Ortiz, N. R., Fault Tree Analysis for Vital Area Identification, June 1979, SAND79-0946, Sandia Laboratories, Albuquerque, NM.

3. Worrell, R. B., and Stack, D. W., A SETS User's Manual for the Fault Tree Analyst, November 1978, SAND77-2051, Sandia Laboratories, Albuquerque, NM.

4. Oliver, D. A., Fault Tree Drawing Program User's Instructions, SLA-73-0409, Sandia Laboratories, Albuquerque, NM.

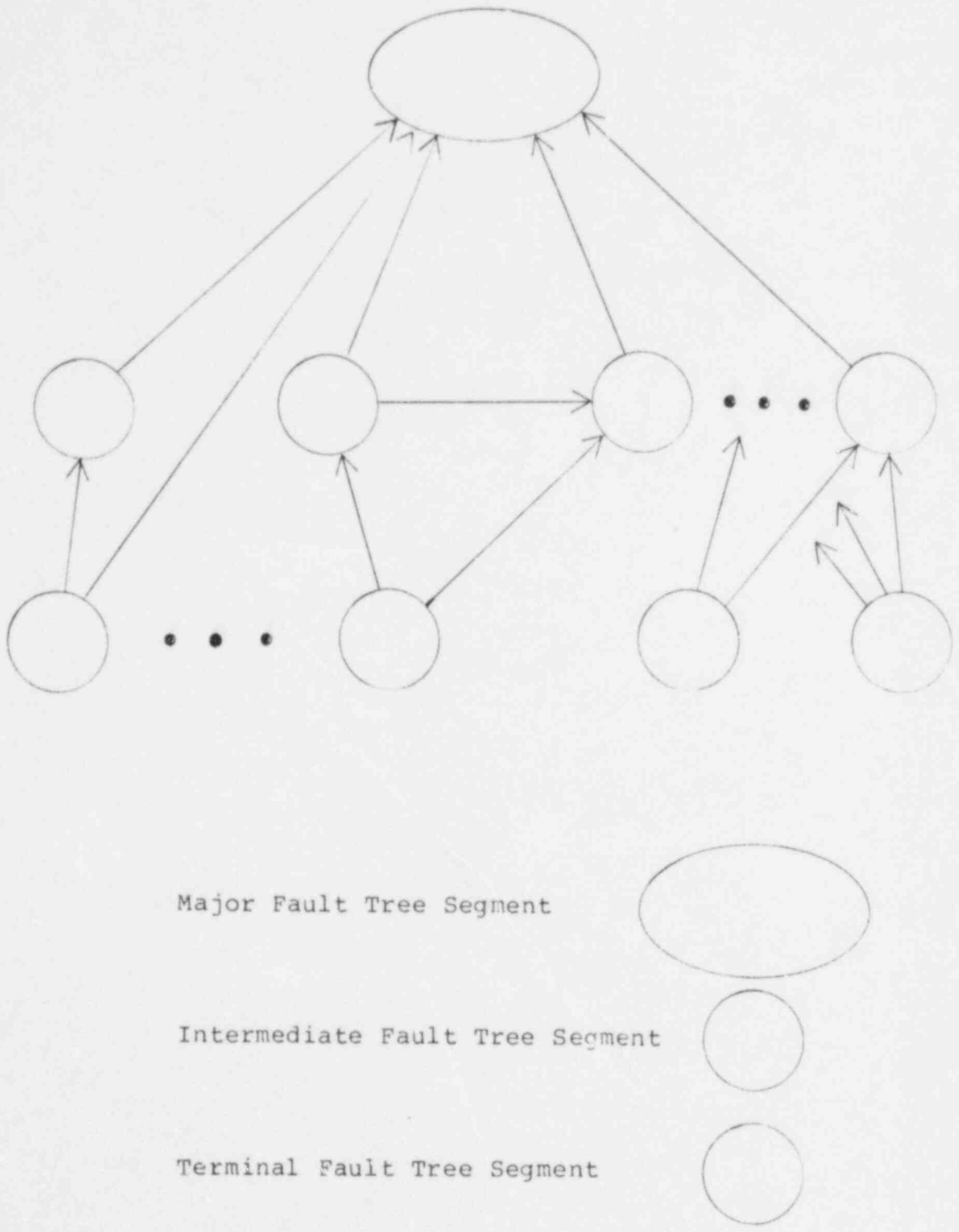

Figure 1. Graphical Representation of Segment Relationships in a Fault Tree Plot.

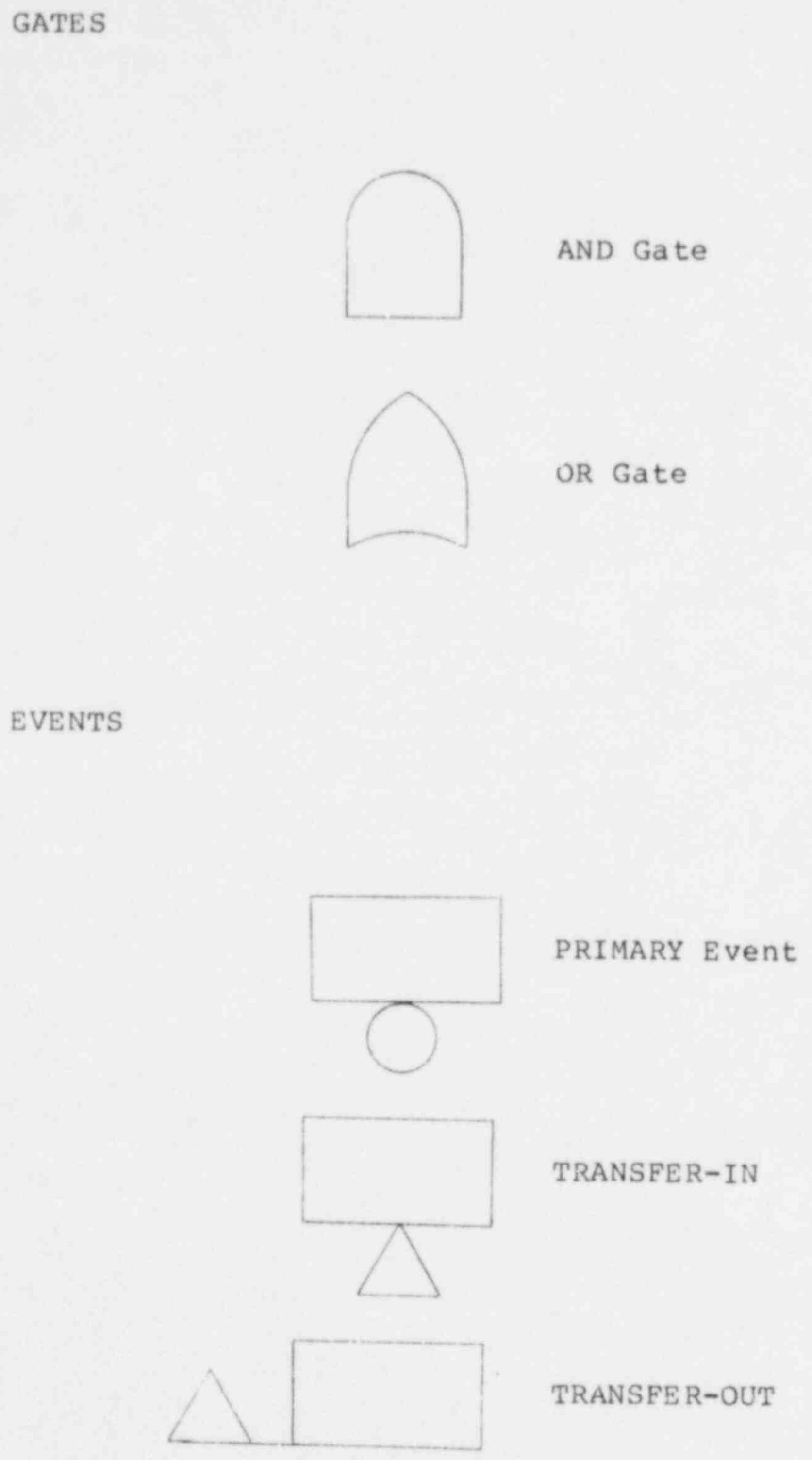

Figure 2. Significant Symbols of the Fault Tree Plot.

Distribution:

U. S. Nuclear Regulatory Commission
(140 Copies for GM)
Division of Document Control
Distribution Services Branch
7920 Norfolk Avenue
Bethesda, MD 20014

1233	M. D. Olman
4410	D. J. McCloskey
4412	J. W. Hickman
4412	S. W. Hatch
4412	A. M. Kolaczkowski
4414	G. B. Varnado
4414	S. L. Daniel
4414	A. W. Frazier
4414	M. S. Hill
4414	D. W. Stack (15)
4414	R. B. Worrell
5642	B. L. Hulme
8266	E. A. Aas
3141	T. L. Werner (5)
3151	W. L. Garner (3)
	For DOE/TIC (Unlimited Release)
	(R. P. Campbell, 3154-3) (25)
	for NRC Distribution to NTIS