An Introduction to Fault Tree Analysis

Prof John Andrews, University of Nottingham

Dr Sally Lunt, University of Nottingham

Key Words: fault trees, failure probability, importance measures, system failure intensity

SUMMARY

A fault tree represents the causes of a specified system failure mode in terms of the failure modes of the system components. The analysis of the fault tree can produce two types of result: qualitative and quantitative. Qualitative results specify the minimal combinations of component failures which result in system failure. Quantification provides the probability or frequency of the system failure mode.

The tutorial will explain the mathematics used to perform a fault tree analysis. A considerable focus of the tutorial will also be on the development of the fault tree model from the engineering system. The techniques are illustrated using a practical example.

1. INTRODUCTION

Fault tree analysis is now a commonly applied method to predict the failure probability or failure frequency of engineering systems in terms of the failure and repair parameters of the system components. The concept of expressing the system failure causes in a logic diagram, which became known as a fault tree, was established in the early 1960's by Watson working at Bell Telephone Labs on the launch control system of the Minuteman intercontinental ballistic missile. The time-dependent methodology to quantify the system failure likelihood or frequency, known as kinetic tree theory was developed almost 10 years later by Vesely [ref 1] working at the Idaho Nuclear Corporation. Enhancements to the technique including the development of importance measures [refs 2,3] and initiator and enabler theory [ref 4] added to the capability. In recent years an alternative to kinetic tree theory for efficient and accurate fault tree quantification has been developed known as the Binary Decision Diagram method [refs 5-11].

Once constructed and appropriate data supplied for the basic events the analysis of the fault tree can be undertaken. Analysis produces two types of result: qualitative and quantitative. Qualitative analysis produces the minimal combinations of basic (component failure) events which result in the system failure mode (top event). These are known as minimal cut sets. Quantitative results include the top event unavailability, unreliability or failure rate. The top event parameters are defines as follows:

unavailability: Qsys(t), the probability that the system failure mode exists at time t

unreliability: Fsys(t) the probability that the system failure mode occurs at least once from time 0 to time t.

failure rate: the rate at which the system failure mode occurs

All of these quantities can be used to judge the acceptability of the system performance. If required the quantification can be extended to produce importance measures which identify the contribution each basic event makes to the top event.

2. FAULT TREE SYMBOLS AND CONSTRUCTION

The features of a typical fault tree are shown in the figure 1.

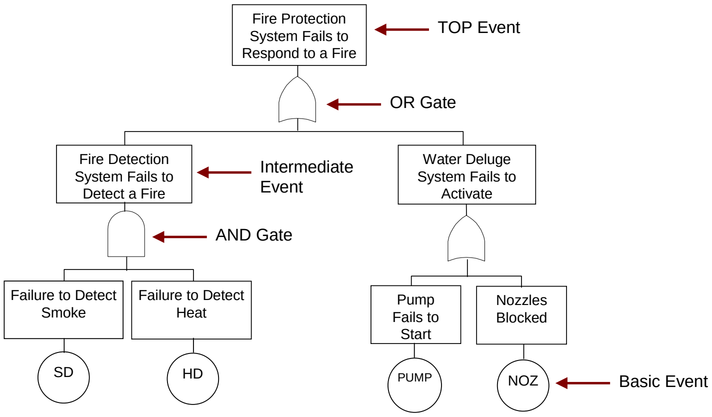


Figure 1. Typical Fault Tree Features

The tree structure starts at the high level system failure mode and progresses in branches spreading downward developing its causality in terms of lower resolution events until component failure modes, basic events, appear. When the lowest resolution events, component failures, are encountered then this defines the limit of the analysis and the development of the failure logic is terminated.

The system failure mode of concern is known for obvious reasons as the 'top event'. Typical examples of this type of event are:

1. total loss of production
2. safety system fails to respond
3. standby system fails to start

4. explosion
5. loss of space mission
6. release of radioactive material

Note that for the first three of these events the system failure can be tolerated and the repair of the causes of the failure will produce the non-occurrence of the top event. For the latter three when the top event has occurred then repairing the component failures which have contributed to its occurrence will not remove the top event.

Typical events which terminate the logic development are:

1. pump fails to start
2. valve fails closed
3. flow sensor fails to indicate high flow
4. operator fails to respond

The first three of these events are hardware failures which specify the piece of equipment which has failed and also the mode in which it fails. Events which do not specify the failure mode at either system or basic event level are unhelpful and should be avoided.

There are two types of symbols which appear in the fault tree structure: gates and events. The events start at a high, system, level at the top of the diagram and progress, through intermediate events, to finer resolution events as you move down the diagram through sub-system and section level down to component level. Typical examples of event symbols used in the fault tree structure are illustrated in figure 2.

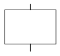
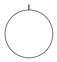
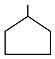
Symbol	Name	Meaning
	Intermediate	System or component event description.
	Basic	Basic event for which failure and repair data is available. Usually represents a component failure.
	House	Represents definitely occurring or definitely not occurring events.

Figure 2. Event Symbols

The events in the fault tree are linked using ‘gate symbols. Common gates are shown in figure 3. The three fundamental logic gates are ‘OR’, ‘AND’ and ‘NOT’. The output (higher level event of an OR gate will result from the occurrence of at least one of the input (lower level) events. For an AND gate the output event occurrence requires the simultaneous existence of all of the input events. The output to a NOT gate happens as long as the input event does not.

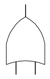
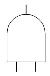
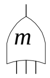
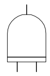
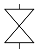
Symbol	Name	Causal Relation
	OR	Output event occurs if at least one of the input events occur.
	AND	Output event occurs if all input events occur.
	VOTE	Output event occurs if at least *m* of the input events occur.
	PRIORITY AND	Output event occurs if all input events occur in sequential order from left to right.
	NOT	Output event occurs if the input event does not occur.

Figure 3. Gate Symbols

Other gates included in figure 3 are the ‘VOTE’ gate where at least m of the inputs has to occur to produce the output event, and the ‘PRIORITY AND’ gate where, like the AND gate, all inputs have to occur but they have to occur in the sequence specified by the list of input events going from left to right.

Sensors used to detect undesirable conditions are frequently arranged in a voting configuration to give a high chance of successfully identifying the condition, but a low chance of a spurious identification when the event does not exist. For example, if it takes 2-out-of-3 sensors to work to correctly detect a hazardous condition for which a system trip will occur (2-out-of-3:W) then one sensor failing to detect the event presence can be tolerated but a second means that there is only one working sensor and the trip condition cannot be satisfied. This system failure is also a 2-out-of-3 voting configuration but this time 2-out-of-3:F. This is represented in the fault tree with a VOTE gate shown in figure 4.

If there are 4 sensors and 2 are required to recognise a condition to trip the system then up to 2 sensor failures can be tolerated. The occurrence of a third sensor failure in this voting configuration leave the system unable to satisfy the 2-out-of-4:W condition and hence will fail. Therefore a 2-out-of-4:W system is a 3-out-of-4:F.

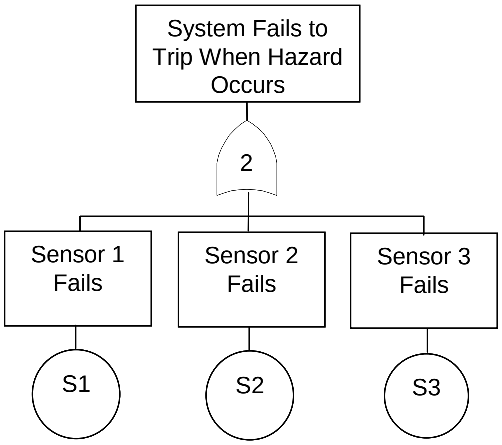


Figure 4. 2-out-of-3:F Vote Gate

The house event, shown in figure 2, is an event which terminates a branch of the fault tree but unlike the basic event, the house event is known to be either true or false. Setting such events to true or false on the fault tree has the effect of turning on or off branches in the fault tree. House events can be used when fault trees are developed for systems which have several operating modes, sections taken out for maintenance, or to represent different design options.

The process to construct a fault tree for a system can be time-consuming and the engineer must have a very thorough understanding of the system before it can take place. Each fault tree explores the causes of one particular system failure mode and therefore it may be necessary to draw more than one fault tree for any system.

Unfortunately there are not a set of rules which can be stated and guarantee the fault tree constructed will have the correct system failure logic. Guidelines [refs 12-14] which help develop a structured and systematic way of generating the fault trees can be given which will provide a process which is less prone to error. These guidelines are:

1. **Assume no miracles:**
   If the normal functioning of a component propagates a fault sequence then it is assumed that the component functions normally. If a component failure fortuitously prevents a fault sequence then this is a miracle and should not be included in the system failure logic development.
2. **Complete-the-gate:**
   Define all inputs to a gate before the further development of any one is undertaken.
3. **No gate-to-gate:**
   Gate inputs should be properly defined and gates should not be directly connected to other gates.

As an example of applying these guidelines to construct a fault tree consider the system, shown in figure 5, designed to react to an undesired gas presence. In the event of a gas leak the system is required to perform two functions. It isolates the sections so that the size of the leak is limited to the inventory contained between the two isolation valves, and de-pressurises the section by flaring the gas. Isolation is achieved by closing two normally open isolation valves. Flaring the gas is achieved by opening the normally closed blowdown valve. For the system illustrated the gas leak is detected by two sensors each of a different type. One (SD1), is a sonic detector, the other (CD1) triggers on gas concentration. The controlling computer will issue a system trip as soon as either of the detectors indicate a gas presence. The computer will automatically drop out a relay which removes power to each of the 3 valves. As a secondary means of achieving the same objective an alarm is sounded which informs the operator of the leak. The operator then activates the push button to de-energise the valves.

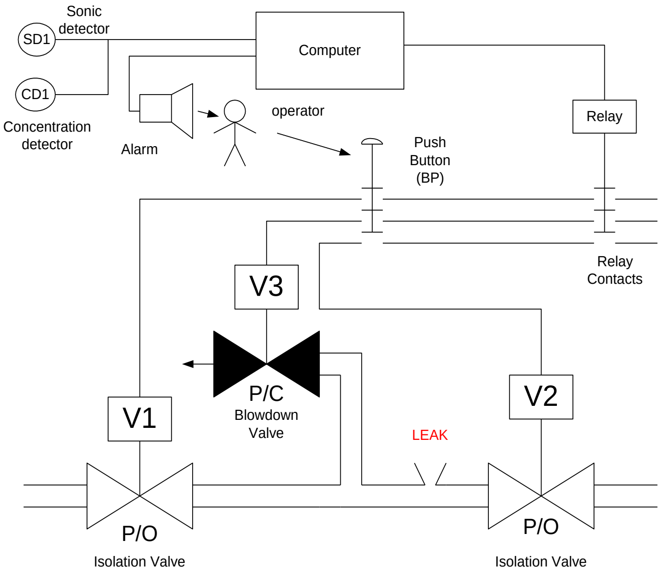


Figure 5. Gas Leak Detection System

The failure modes to consider for each of the components in the system are given in Table 1.

Component failure mode	code
Isolation valve 1 fails to close	V1
Isolation valve 2 fails to close	V2
Blowdown valve 3 fails to open	V3
Operator unavailable	OP
Computer fails to process trip condition	COMP
Alarm fails to sound	AL
Relay contacts stuck closed	CONT
Concentration detector fails to register leak	CD1
Sonic detector fails to register leak	SD1
Push Button contacts stuck closed	PB

Table 1. Component Failure Modes

Given a gas leak the system should perform three tasks:

- close isolation valve V1
- close isolation valve V2
- open blowdown valve V3

A fault tree has been drawn for the Top Event 'leak detection system fails'. This is shown in figure 6.

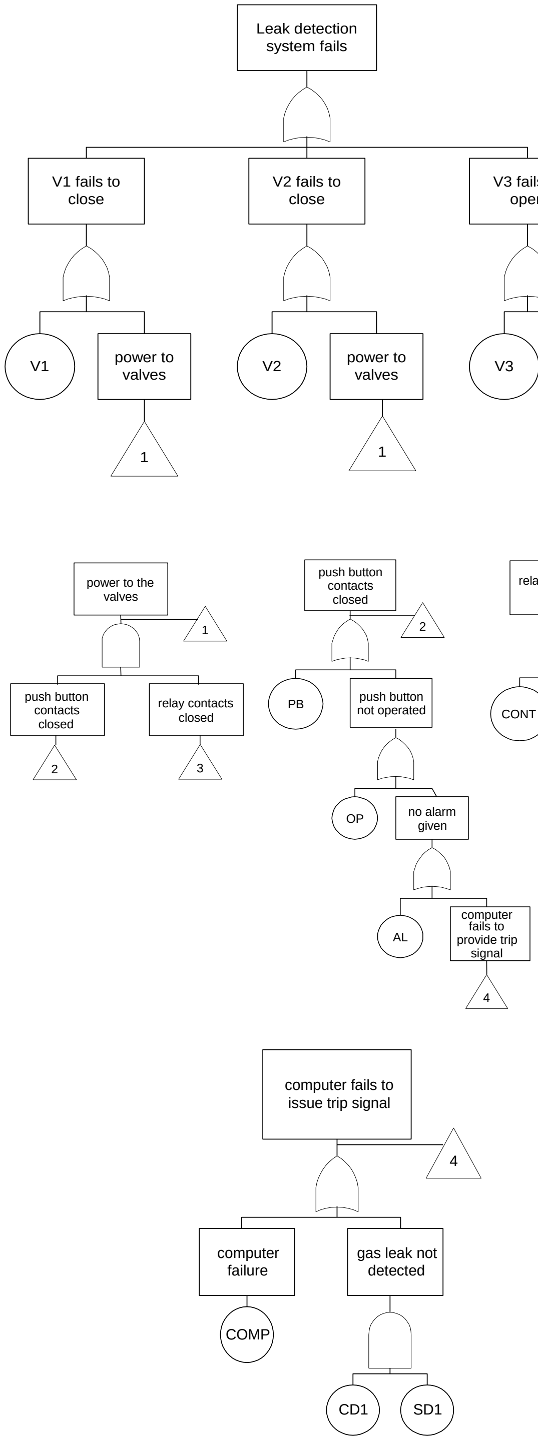


Figure 6. Gas Detection System Fault Tree

3. MINIMAL CUT SETS

A system failure analysis using a fault tree can establish the component conditions that will yield a system failed state. A list of component failed states which cause the system failure mode is known as a cut set. This information is however not that useful as there can be component failures included in the list which are not needed to cause the system failure since other component failures will have already guaranteed that the system will fail. Removing these redundant component failure events from the list gives *minimal cut sets.* Minimal cut sets are a list of minimal (necessary and sufficient) component failed states which cause the system failure mode.

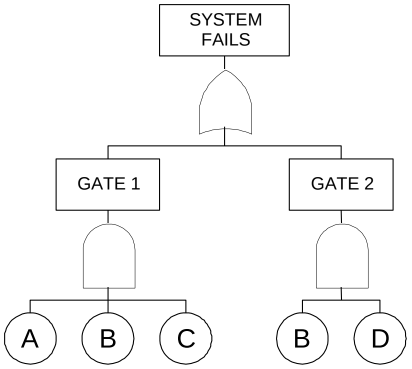


Figure 7. Example System Fault Tree Structure.

By inspection the minimal cut sets of the fault tree shown in figure 7 are: {A,B,C} and {B,D}. The way that the fault tree represents the system failure logic is not unique and different engineers will probably draw a different tree structure for the same system failure mode. Whilst the actual diagram structures may be different, if they represent the same logic function, they will produce the same minimal cut sets.

To produce the minimal cut sets of a fault tree a Boolean equation is established for the Top Event which is then manipulated into its minimal sum-of-products form (disjunctive normal form) to enable the minimal cut sets to be identified.

A Boolean variable is defined for each basic event which is TRUE if the basic event occurs and FALSE if it does not. As an example consider the fault tree in figure 8.

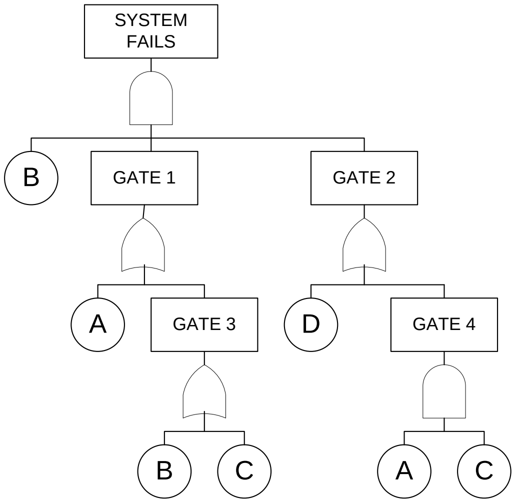


Figure 8. Example Fault Tree

Using a top-down approach we get the following Boolean expression for the top event in terms of the component failure conditions:

TOP=B.GATE1.GATE2
=B.(A+GATE3).(D+GATE4)
=B.(A.D+A.GATE4+ GATE3.D+GATE3.GATE4)
=B.[A.D+A.A.C+(B+C).D+(B+C).A.C]
=B.[A.D+A.A.C+B.D+C.D+B.A.C+C.A.C]
(1)

Where '.' represents AND and '+' represents OR in the equations. These equations are then simplified using the laws of Boolean Algebra**:**

Idempotent	**A.A=A**	(1) removes repeated events within each cut set
	A+A=A	(2) removes repeated cut sets from the expression
Absorption	**A+A.B =A**	(3) removes non-minimal failure combinations

Applying idempotent rule (1) to equation 1 gives:
TOP=B.[A.D+A.C+B.D+C.D+B.A.C+C.A] (2)

Applying rule (2) gives:
TOP=B.[A.D+A.C+B.D+C.D+B.A.C] (3)

Applying rule (3) gives:
TOP=B.[A.D+A.C+B.D+C.D] (4)

Expanding out and applying these rules further gives:
TOP=B.D+A.B.C (5)

This form of the equations is in its simplest sum-of-products form and cannot be reduced any further. The products of this expression are the minimal cut sets. Therefore the fault tree shown in figure 8 has minimal cut sets: **B.D** and **A.B.C** (showing the fault tree to be equivalent to that shown in figure 7).

4. COMPONENT FAILURE PROBABILITY

To quantify the fault tree the component mode failure probabilities must be predicted. The models used to make this prediction depend on how the component is maintained and three situations are considered here: no repair, repair when the failure occurs (revealed failure), and repair when the failure is discovered (unrevealed failures).

No Repair

When a component cannot be repaired then its chance of failure will continue to increase over time to its limiting value of 1 as shown in Figure 9.
In such circumstance if the component is functioning at a time t then it must have worked continuously to that time and so its reliability and availability are the same. Therefore the unreliability, F(t), and unavailability, Q(t), are the same and if the component has a constant failure rate, λ, these are given by:

$$Q(t)=F(t)=1-e^{-\lambda t}\le \lambda t \qquad (6)$$

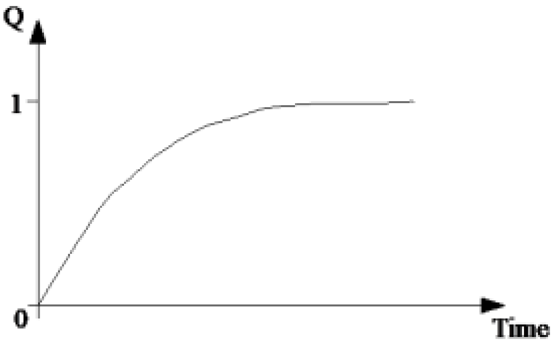


Figure 9. Non-repairable Component

Revealed Failures

It is known when a revealed component failure occurs and the repair can be started immediately. This is unscheduled maintenance which takes place in response to the component failure occurrence. For components with constant failure rate, λ, and constant repair rate, υ, the unavailability at time t (illustrated in figure 10) is given by:

$$Q(t)=\frac{\lambda}{\lambda+\nu}\left(1-e^{-(\lambda+\nu)t}\right) \qquad (7)$$

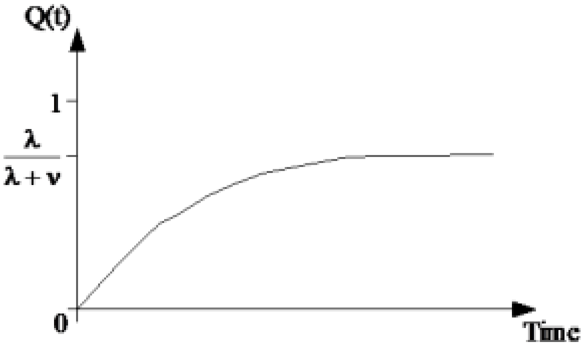


Figure 10. Revealed Component Failure

Note that when the times to an event are given by the exponential distribution and occur with a constant rate then the mean time to the event is 1/rate so:

Mean time to failure, μ = 1/ λ
and Mean time to repair, τ=1/ υ

Unrevealed failure

When components are part of standby or safety systems which only operate under certain conditions then when failures occur they will not be noticed. For this type of system they must be tested to reveal the failure and so the repair takes place when scheduled tests are carried out. This results in the failure probability distribution shown in figure 11.

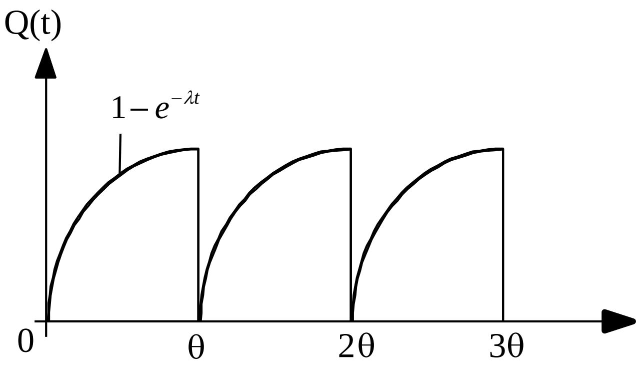


Figure 11. Unrevealed Component Failure

The average unavailability is given by:

$$
\begin{aligned}
Q_{AV} &= \frac{1}{\theta}\int_0^{\theta}(1-e^{-\lambda t})dt \\
&= \frac{1}{\theta}\left[t+\frac{e^{-\lambda t}}{\lambda}\right]_0^{\theta} \\
&= 1-\frac{\left(1-e^{-\lambda\theta}\right)}{\lambda\theta}
\end{aligned}
\tag{8}
$$

Where θ is the interval between inspections. Alternatively this can be approximated by:

$$Q_{AV} = \lambda\left(\frac{\theta}{2}+\tau\right) \tag{9}$$

5. MINIMAL CUT SET FAILURE PROBABILITY

Assuming the components fail independently of each other the calculation of the minimal cut set, C_i, probabilities is trivial and given by:

$$P(C_i) = \prod_{j=1}^{n} P(X_j) \tag{10}$$

where the events in the minimal cut set, C_i, are X_1, X_2, … X_n.

6. SYSTEM FAILURE PROBABILITY

Using fault tree analysis predictions for the failure probability or the failure frequency of the system (top event) can be made. In this section we will concentrate on the top event probability. Having obtained the minimal cut sets we can express the top event logic equation as the disjunction (OR) of the N_C minimal cut sets, C_i. The system failure probability, Q_{sys}, is then the probability of this disjunction:

$$
\begin{aligned}
T &= C_1 + C_2 + \cdots + C_{N_C} \\
Q_{SYS} &= P(T) = P(C_1 + C_2 + \cdots + C_{N_C})
\end{aligned}
\tag{11}
$$

Then top event probability is then evaluated using the inclusion-exclusion expansion:

$$
\begin{aligned}
Q_{SYS} = {} & \sum_{i=1}^{N_C} P(C_i) - \sum_{i=2}^{N_C}\sum_{j=1}^{i-1} P(C_i \cap C_j) + \\
& \sum_{i=3}^{N_C}\sum_{j=2}^{i-1}\sum_{k=1}^{j-1} P(C_i \cap C_j \cap C_k) - \cdots \\
& \cdots + (-1)^{N_C+1} P(C_1 \cap C_2 \cdots \cap C_{N_C})
\end{aligned}
\tag{12}
$$

Consider the example fault tree in figure 7 which has minimal cut sets {B,D}, {A,B,C}. Applying equation 12 gives:

$$Q_{SYS} = q_a q_b q_c + q_b q_d - q_a q_b q_c q_d \tag{13}$$

where q_A, q_B, q_C, q_D are the failure probabilities of components A, B, C and D respectively.

In this particular example it is a simple calculation. However, consider a moderate to large sized fault tree which delivered 100,000 minimal cut sets. The number of elements in first term of equation 12 would be10^5, in the second term ≈ 5 x 10^9 and in the third term ≈ 1.7 x 10^{14} and so on for the10^5 terms in the equation. Even with modern fast digital computers this is an enormous number of calculations and would take a considerable time to complete. In practice acceptably accurate upper bound approximations are used such as the Rare Event approximation (equation 14) or the Minimal Cut Set Upper Bound (equation 15).

$$Q_{EXACT} \le \sum_{i=1}^{N_C} P(C_i) \tag{14}$$

$$Q_{EXACT} \le 1 - \prod_{i=1}^{N_C}\left(1 - P(C_i)\right) \tag{15}$$

7. IMPORTANCE MEASURES

Should a system not perform to the reliability or availability target required then modifications to the design or operation have to be made to address the weaknesses. An output from a fault tree analysis which can help to identify the weaknesses is importance measures. Importance measures provide an indication, in some sense, of the contribution that each basic event or minimal cut set makes to the system failure mode. There are many different types of importance measure and each calculates a different means of ranking the contribution to the top event. More details can be found in references 12 and 14. Considering the basic event importance measures. The vulnerability of the system to the occurrence of each component failure event is indicated by a numerical value. The higher the importance value the greater the contribution of that basic event to the system failure. Depending on nature of the importance measure they can take into account such things as the structure of the system (levels of redundancy etc), the failure rate of the component, and the time taken to repair the component. To improve the system performance the basic events which have the highest importance measure can be addressed. Importance measures can be deterministic – which consider only the system structure or probabilistic and account for the likelihood of component failures.

A concept which is fundamental in developing component importance measures is that of a critical system state.

A ***Critical System State*** for a component i is a state for the remaining n-1 components such that failure of component i causes the system to go from a working to a failed state.

Structural Measure of Importance

Having defined the critical system states the **structural measure of importance**, I_i, can be defined:

$$I_i = \frac{\text{number of critical states for component i}}{\text{total number of states for the (n - 1) remaining components}} \qquad (16)$$

Consider a simple system of 4 components whose failure causes are represented by the fault tree in figure 12. Where the failure of the components are given by: $q_A = q_C = 0.1$, and $q_B = q_D = 0.2$.

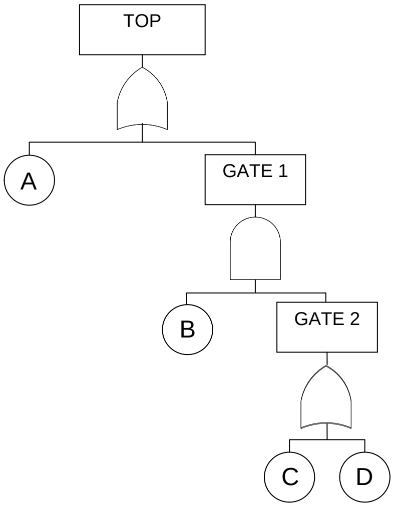


Figure 12. Simple Four Component System Fault Tree

Taking each component in turn the critical system states can be identified by constructing a table which considers the states of all the other components in the system. Some of these states may already satisfy the conditions which mean the system is failed. Others will mean that the system still functions. From these states, those which will fail the system when the component being considered fails are critical and identified. These tables are illustrated for components A, B and C in tables 2, 3 and 4 respectively. Due to the symmetry of the system component D will have the same number of critical states as component C.

	States			
	B	C	D	Critical for A?
1	W	W	W	Y
2	W	W	F	Y
3	W	F	W	Y
4	W	F	F	Y
5	F	W	W	Y
6	F	W	F	N
7	F	F	W	N
8	F	F	F	N

Table 2. Criticality of Component A

	States			
	A	C	D	Critical for B?
1	W	W	W	N
2	W	W	F	Y
3	W	F	W	Y
4	W	F	F	Y
5	F	W	W	N
6	F	W	F	N
7	F	F	W	N
8	F	F	F	N

Table 3. Criticality of component B

	States			
	A	B	D	Critical for C/D?
1	W	W	W	N
2	W	W	F	N
3	W	F	W	Y
4	W	F	F	N
5	F	W	W	N
6	F	W	F	N
7	F	F	W	N
8	F	F	F	N

Table 4. Criticality of Component C

This gives structural importance measures for the components of:

$$I_A = 5/8$$
$$I_B = 3/8 \qquad (17)$$
$$I_C = I_D = 1/8$$

Birnbaum Measure of Importance

The ***Criticality Function***, $G_i(q)$, is the probability that the system is in a critical state for component i . This is also known as ***Birnbaum's measure of importance.***

From table 2 Birbaum's measure of importance for component A is given by summing the probability of being in a critical state. This is:

$$\begin{aligned} G_A &= (1 - q_B)(1 - q_C)(1 - q_D) \\ &\quad + (1 - q_B)(1 - q_C)\, q_D \\ &\quad + (1 - q_B)(\, q_C)(1 - q_D) \\ &\quad + (1 - q_B)\, q_C\, q_D + q_B\,(1 - q_C)(1 - q_D) \\ &= (1 - q_B) + q_B\,(1 - q_C)(1 - q_D) \end{aligned} \qquad (18)$$

$G_A = 0.944$

Similarly from tables 3 and 4 we get:

$$\begin{aligned} G_B &= (1 - q_A)(1 - q_C)\, q_D \\ &\quad + (1 - q_A)\, q_C\,(1 - q_D) \\ &\quad + (1 - q_A)\, q_C\, q_D \end{aligned} \qquad (19)$$

$G_B = 0.252$

and

$$G_C = (1 - q_A)\, q_B\,(1 - q_D) \qquad (20)$$

$G_C = 0.144$

$$G_D = (1 - q_A)\, q_B\,(1 - q_C) \qquad (21)$$

$G_D = 0.162$

Whilst the structural and Birnbaum measures can be produced using the tabular approach this soon becomes impractical for real systems due to the size of the tables.

An alternative means of calculating Birnbaum's measure is to use:

$$G_i(\underline{q}) = \frac{\partial Q_{sys}}{\partial q_i} = Q_{sys}(1_i, \underline{q}) - Q_{sys}(0_i, \underline{q}) \qquad (22)$$

where $Q_{sys}(1_i,q)$ is the system failure probability with $q_i=1$ and $Q_{sys}(0_i,q)$ is the system failure probability with $q_i=0$.

Criticality Measure of Importance

The criticality measure of importance for component i is the contribution to the system failure probability due to the system being in a critical state for component i and i failing.

$$I_{CM\,i} = \frac{G_i(\mathbf{q}(t))q_i(t)}{Q_{SYS}(t)} \qquad (23)$$

The failure probability of the simple system shown in figure 12, with minimal cut sets {A}, {B,C} and {B,D}, is given by:

$$\begin{aligned} Q_{SYS} &= q_A + q_Bq_C + q_Bq_D - q_Aq_Bq_C - q_Aq_Bq_D - q_Bq_Cq_D + q_Aq_Bq_Cq_D \\ &= 0.1 + 0.02 + 0.04 - 0.002 - 0.004 - 0.004 + 0.0004 \\ &= [0.16] - [0.01] + [0.0004] = 0.1504 \end{aligned} \qquad (24)$$

The criticality importance measures for the components are then:

$$\begin{aligned} I_{CM_A} &= \frac{(0.944)(0.1)}{0.1504} = 0.6277 \\ I_{CM_B} &= \frac{(0.252)(0.2)}{0.1504} = 0.3351 \\ I_{CM_C} &= \frac{(0.144)(0.1)}{0.1504} = 0.0957 \\ I_{CM_D} &= \frac{(0.162)(0.2)}{0.1504} = 0.2154 \end{aligned} \qquad (25)$$

Fussell -Vesely Measure of Importance

The Fussell-Vesely measure of component importance for component i is defined as the ratio of the probability of the union of all minimal cut sets containing i and the system failure probability.

$$I_{FV_i} = \frac{P\left(\bigcup_{i \in C_j} C_j\right)}{Q_{SYS}} \qquad (26)$$

For the simple system shown in figure 12 this measure gives:

$$I_{FV_A} = \frac{q_A}{Q_{SYS}} = \frac{0.1}{0.1504} = 0.6649$$

$$\begin{aligned} I_{FV_B} &= \frac{q_B(q_C + q_D - q_Cq_D)}{Q_{SYS}} \\ &= \frac{0.2(0.1 + 0.2 - 0.02)}{0.1504} = 0.3723 \end{aligned} \qquad (27)$$

$$I_{FV_C} = \frac{q_Cq_B}{Q_{SYS}} = \frac{0.02}{0.1504} = 0.1330$$

$$I_{FV_D} = \frac{q_Dq_B}{Q_{SYS}} = \frac{0.04}{0.1504} = 0.2660$$

8. SYSTEM FAILURE INTENSITY

Let $w_{SYS}(t)$ be the system failure intensity at time t. Having calculated Birnbaum's measure of importance for each of the n components means that the system failure intensity can be determined from:

$$w_{SYS}(t) = \sum_{i=1}^{n} G_i(\underline{q}).w_i(t) \qquad (28)$$

where w_i is the component failure intensity and $G_i(q)$ is the Criticality Function.

9. SYSTEM CASE STUDY

As an example of applying a fault tree analysis to a system consider the simple tank level control system shown in figure 13. Initially the system has the push button contacts open and switches 1 and 2 (SW1, SW2) contacts closed. To start the system the push button is pressed and held. This energises relay R1 which closes its contacts and maintains the circuit when the push button is released. Relay R2 is also energised and its contacts close, starting the pump in the second circuit. The pump transfers fluid to the tank. The level of the tank fluid is monitored by two level sensors L1 and L2. When the tank fluid reaches the required level switch SW1 opens and de-energises relay R2 turning off the pump. When the fluid in the tank is used and the level drops SW1 will close and pump fluid to replace that used. The normal operation of the system is the switch SW1 opening and closing which turns off and on the pump.

As a safety feature, the second level sensor, L2, is connected to switch SW2. When the fluid level is unacceptably high SW2 opens which de-energises relay R1. R1 contacts then open to break the control circuit. This results in R2 de-energising, its contacts open and remove power from the pump. This will require a manual start-up of the circuit.

For the system failure mode ‘ Tank overfills’, the relevant component failure modes along with the failure rate and repair time data are shown in table 5. Some of the failure modes will be revealed such as relay R2 contacts stuck closed. This component condition will mean that the pump keeps running and the problem is revealed by the tank overfilling. Others such as relay R1 contacts fail closed will be unrevealed as this is the normal operating state for that component. All of the component failure modes associated with the safety systems (L2, SW2, R1 and PB) will be unrevealed as for this class of events the failure will only be revealed when the component is tested /inspected or when a demand for the component to work occurs. For these component failure events an inspection interval is also specified which enables the probability of the event to be calculated. For this example an inspection interval of 4380 hours is assumed.

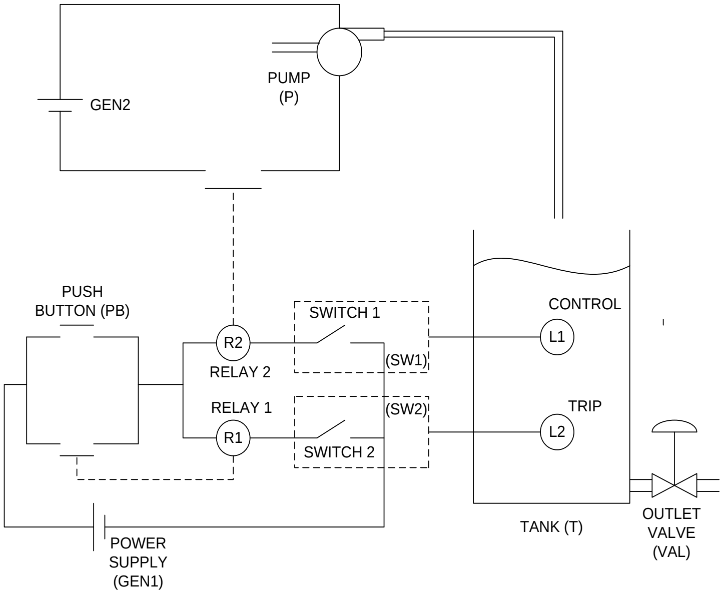


Figure 13. Simple Tank Level Control System

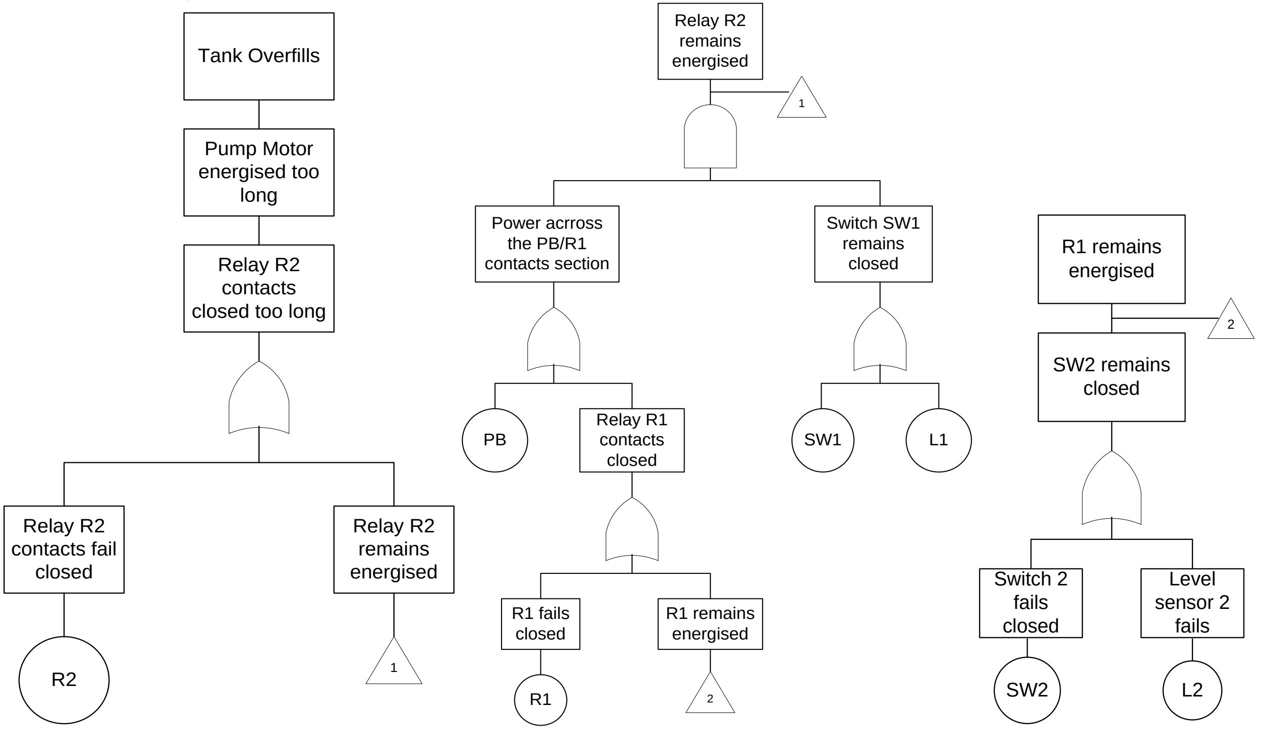

Tank Overfills
Pump Motor energised too long
Relay R2 contacts closed too long
Relay R2 contacts fail closed
R2
Relay R2 remains energised
1
Relay R2 remains energised
1
Power acrross the PB/R1 contacts section
PB
Relay R1 contacts closed
R1 fails closed
R1
R1 remains energised
2
Switch SW1 remains closed
SW1
L1
R1 remains energised
2
SW2 remains closed
Switch 2 fails closed
SW2
Level sensor 2 fails
L2

Figure 14. Fault Tree for Top Event ‘Tank Overfills’

Component	Failure Mode	Code	Failure Rate (per hour)	Mean Time to Repair (hours)
Push Button	Stuck closed	PB	5. x 10^{-5}	2.
Relay Contacts	Stuck closed	R1/R2	6. x 10^{-5}	10.
Switch	Stuck closed	SW1/SW2	5. x 10^{-5}	10.
Level Sensors	Fail to indicate high level	L1/L2	2. x 10^{-6}	5.

Table 5. Component Failure Modes and Data

The fault tree for the undesired top event ‘ tank overfills’ is developed in figure 14.

The text boxes specify exactly what each gate output event in the fault tree represents. Each branch is developed downward using AND and OR gates until basic events (component failure events) are encountered and the failure causality development is terminated.

The final fault tree structure showing how the basic events combine to cause the system level failure event is illustrated in figure 15

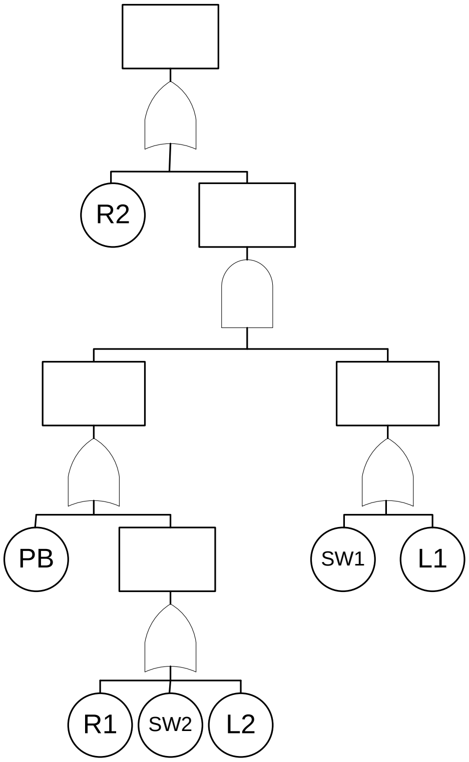


Figure 15. Tank Overfill Fault Tree Structure

For the tank level control system fault tree the complete list of minimal cut sets are given in table 6. As can be seen there are 9 failure combinations in total. One is first order (a single event causes system failure) and eight are of order two.

1	R2	
2	SW1	PB
3	SW1	R1
4	SW1	SW2
5	SW1	L2
6	L1	PB
7	L1	R1
8	L1	SW2
9	L1	R1

Table 6. Minimal Cut Sets

Using the component failure data in table 1, the system failure parameters can be calculated:

Top Event Probability = 1.39 x 10^{-3}
Top Event Frequency = 1.919 x 10^{-4} per hour

If the system failure predictions indicate an unacceptable performance the weaknesses can be identified using component importance measures. The Fussell-Vesely measure is indicated in table 7. This shows that component L1 provides the biggest contribution to system failure.

Rank	***Component***	***Fussell-Vesely***
1	**L1**	**0.4148**
2	**R2**	**0.3777**
3	**L2**	**0.3155**
4	**SW1**	**0.2075**
5	**R1**	**0.1139**
6	**SW2**	**0.0966**
7	**PB**	**0.0963**

Table 7 Importance Measures

The system assessment results presented have been obtained using a commercial software package.

10. CONCLUSIONS

A fault tree represents the causes of a specified system failure mode in terms of the failure modes of the system components. A summary of the features of fault tree analysis is:

- Provides a well structured development of the system failure logic.

- Forms a documented record of analysis which can be used to communicate fault development with regulators etc.
- Directly developed from the engineering system structure.
- Easily interpreted from the engineering viewpoint.
- Analysis gives all minimal cut sets.
- Quantification gives the top system failure mode probability or frequency.
- Vulnerability to system failure can be identified using importance measures.

11. REFERENCES

1. W.E. Vesely, ‘A Time Dependent Methodology for Fault Tree Evaluation’, Nuclear Design and Engineering, no. 13 (1970): 337-360.
2. Z.W.Birnbaum, ‘On the importance of different components in a multi-component system’, Multivariate Analysis 11, P.R.Krishnaiah, ed.,Academic Press, 1969
3. Fussell, J. B., ‘How to Hand-Calculate System Reliability Characteristics’, IEEE Transactions on Reliability, R-24, (3), 1975
4. Lambert H.E and Dunglinson C., ‘Interval Reliability for Initiating and Enabling events’, , IEEE Transactions on Reliability, Vol 32, June 1983, pp 150-163.
5. Akers B, ‘Binary Decision Diagrams’, IEEE Trans on Computers, 27(6), 509-516, 1978.
6. Bryant R, ‘Graph Based Algorithms for Boolean Function Manipulation’, IEEE Trans on Computers, 35(8), 677-691, 1986.
7. Schneeweiss W., ‘Fault Tree Analysis Using Binary Decision Diagrams’, IEEE Trans on Reliability, 34(5), 453-457, 1985.
8. Rauzy A, ‘New Approaches for Fault Tree Analysis’, Reliability Engineering and System Safety, 05(59), 203-211, 1993.
9. Sinnamon R.M. and Andrews J.D., ‘Quantitative Fault Tree Analysis Using Binary Decision Diagrams’, European Journal of Automation, 30 (8), 1996, 1051-1071.
10. Sinnamon R.M and Andrews J.D., ‘Improved Efficiency in Qualitative Fault Tree Analysis’, Quality and Reliability Engineering International, Vol 13, 1997, pp293-298.
11. Sinnamon R.M and Andrews J.D., ‘Improved Accuracy in Quantitative Fault Tree Analysis’, Quality and Reliability Engineering International, Vol 13, 1997, pp285-292
12. Andrews J.D. and Moss T.R., ‘Reliability and Risk Assessment’, Professional Engineering Publications Ltd, 2002.
13. Haasl D.F., Roberts N.H., Vesely, W.E. and Goldberg F.F., ‘Fault Tree Handbook’, US Nuclear Regulatory Commission NUREG-0492, 1981
14. Henley E.J. and Kumamato H., ‘Reliability Engineering and Risk Assessment’, Prentice-Hall, 1981

BIOGRAPHIES

John Andrews, Ph.D, FIMechE, CEng, MIMA, CMath, MSaRS
Professor of Infrastructure Asset Management
Head of the Resilience Engineering Research Group
University of Nottingham
Faculty of Engineering,
University Park
Nottingham, NG7 2RD, England

email: john.andrews@nottingham.ac.uk

John Andrews is Professor of Infrastructure Asset Management in the Faculty of Engineering at the University of Nottingham, UK. He is also the Head of the Resilience Engineering Research Group. He moved to Nottingham in 2009 having previously worked for 20 years at Loughborough University. The focus of his research has been on methods for predicting system reliability and availability in terms of the component failure probabilities and a representation of the system structure. Much of his early work has concentrated on the Fault Tree technique and the use of the Binary Decision Diagrams (BDDs) as an efficient and accurate solution method. More recently his main interest has been on modelling the effects of maintenance in order to identify the optimal strategy for asset management. He is the author of around 350 research papers on this topic and is joint author, with Bob Moss, of a text book, Reliability and Risk Assessment, now in its second edition, published by ASME. John was the founding Editor of the Journal of Risk and Reliability and is a member of the Editorial Boards for Reliability Engineering and System Safety, and Quality and Reliability Engineering International.

Sally Lunt, BSc, Ph.D
Research Fellow in Risk and Reliability Engineering
Resilience Engineering Research Group
University of Nottingham
Faculty of Engineering,
University Park
Nottingham, NG7 2RD, England

email: sally.lunt@nottingham.ac.uk

Sally Lunt is a Research Fellow at the University of Nottingham. She graduated in Mathematical Education from Loughborough University and then went on to study her doctorate in the Risk and Reliability Engineering Group at the University. The subject of her thesis was importance measure for non-coherent fault trees. Sally has spent a significant proportion of her career to date in education. She recently returned to research with the Resilience Engineering Research Group and specializes in advanced methods for fault tree analysis and phased mission modelling.