Computerised fault tree construction for a train braking system

Computerised Fault Tree Construction for A Train Braking System

J J Henry and J D Andrews

Department of Mathematical Sciences
Loughborough University of Technology
Loughborough, Leicestershire LE11 3TU

A new approach for fault tree automation is proposed which is a hybrid of the digraph and decision table methods, using the best features of both. The new method is based on the flexibility of the decision table method but incorporates a way of detecting, classifying and analysing control loops, similar to the use of operators in the digraph approach. As well as using operators to deal with control loops a new operator is introduced that deals with electrical circuits. This means that when constructing the fault trees, difficulties of handling repeated events are eliminated and the size of the fault trees is significantly reduced. The method has been tested by its application to a braking system on a train.

1. Introduction

One of the most popular approaches to determine the frequency of occurrence of hazardous events is provided by Fault Tree Analysis[1]. The application of this method results in a tree structured diagram representing how the component failures, human actions and software errors contribute to a specified system failure mode. By providing data for the basic events the tree can be analysed to yield the minimal cut sets and system failure probability or frequency.

Unfortunately the development process for fault trees is a very time intensive activity. Fault trees for the WASH 1400 nuclear reactor study[2] took many man years of effort to construct. Also, since there are no rigorous rules which, if applied to a system, guarantee the generation of the correct fault tree, the quality of trees produced manually is very dependent on the experience and abilities of the engineer who has produced them. These two factors, time and quality, have been the main driving forces behind the research into ways in which fault tree construction can be implemented on a computer. To date there have been many attempts to develop such a code. The most successful have been based on digraph approaches[3–5], decision tables[6], expert systems[7] and functional equations[8–12]. Despite the amount of work carried out in this area a commercial package has yet to become available. Deficiencies are reported in the literature for these methods[13].

Due to the historical development of the risk analysis subject the majority of work on fault tree synthesis has been applied to chemical processing plant. The nature of such plant, and in particular their process control systems, means that it does not facilitate accurate fault tree development. The transport industries are now heavily involved in this area. Whilst deficiencies exist in current approaches to automated fault tree construction when applied to process plant, the characteristics of process plant and transport systems are very different. As such the development of an automatic fault tree construction algorithm which can be applied to transport systems may be a more successful proposition.

An alternative method suitable for railway safety systems is presented in reference 14. This paper briefly reviews this method and describes how it has been applied to a train braking system.

2. Overview of The Fault Tree Construction Program

The data flow chart showing the overall package is shown in figure 1. The system description is entered as a system schematic using the facilities in Auto Cad. From the drawing a file is created that holds the topology information for the system. This includes information on which ports of the components are connected together. There is a component decision table library that holds all decision tables for the most common components, any of these decision tables can be amended or new decision tables added using the generic component editor. A top event is now entered into the fault tree construction program AFTCC. The program produces a fault tree and writes this to a formatted file which can directly input to an analysis package. Qualitative and quantitative analysis can then be performed giving the top event probability or frequency of occurrence.

3. Train Emergency Braking System

The braking system (figure 2) is designed so that all the safety functions (tripcock, deadman, passenger and passenger alarm) are monitored by a series of wires running the length of the train. The activation of any of the safety devices will cause an open circuit condition which will, in turn, cause the brakes to be applied on the whole train. The wires are known as the Train Safety Wires. To admit air to, and release air from the brake cylinders, a valve is fitted to each car. This valve is known as the 7 Step Valve as it can vary the pressure supplied to the brake cylinders in seven different steps. In order to keep the brakes released on the car the 7 Step Valve must remain energised. Loss of supply current will cause the valve to admit air pressure to the brake cylinders. The current to the valve is dependant on there being current in the Train Safety Wires.

A feature of the system is the Load-Weight control of braking. This means that the weight of each car is automatically monitored as passengers board and disembark and the amount of air pressure admitted to the brake cylinders during application is adjusted according to the load in the car. This system uses Main Line air and is controlled by a Variable Load Control Valve and a Variable Load Valve mounted next to the 7 Step Valve on each car. If there is a loss of air pressure to the Variable Load Valve, the valve will allow pressure to be supplied to the 7 Step Valve equivalent to tare load.

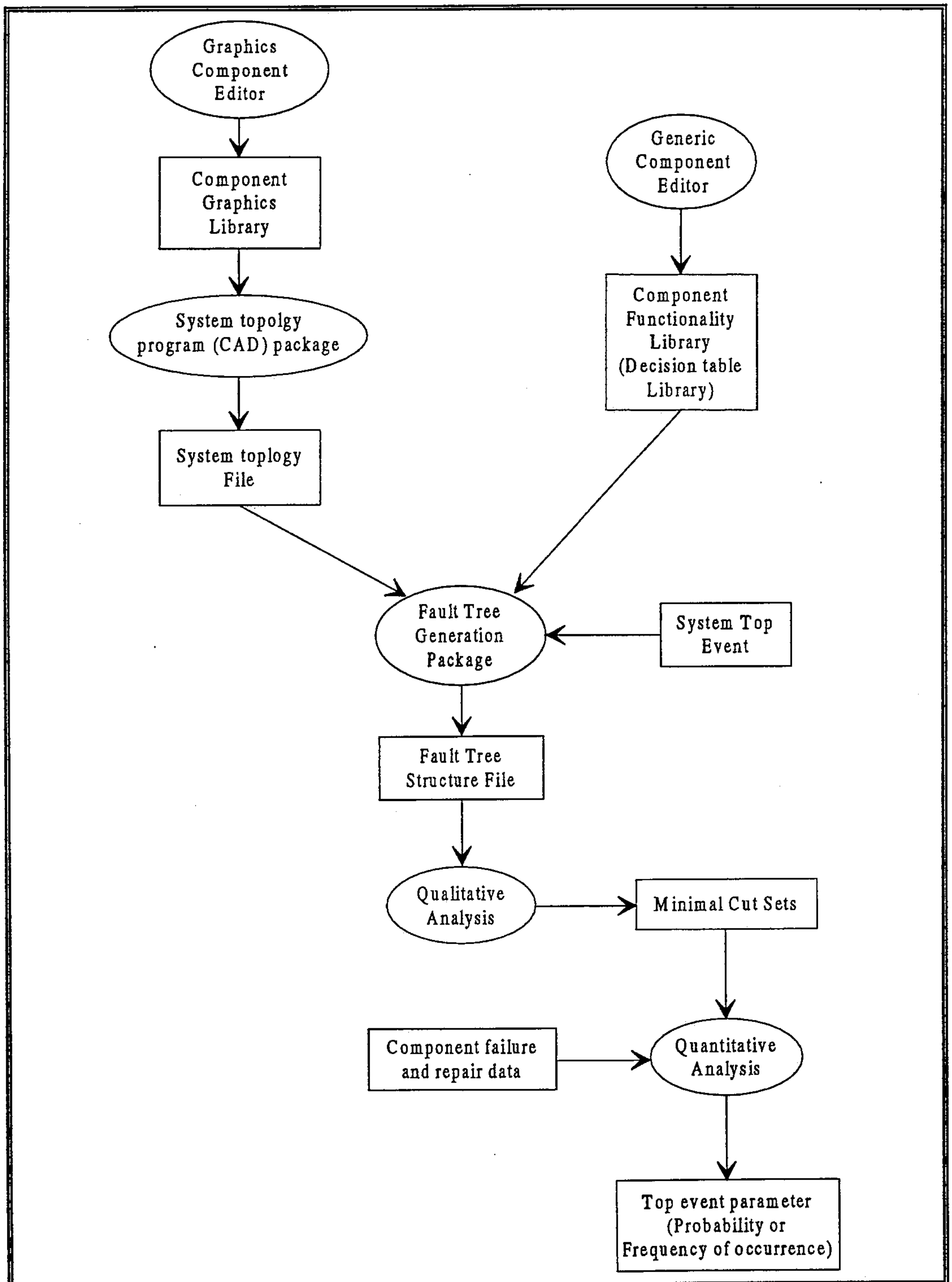

Figure 1 Data Flow Chart

There are three main control circuits for the operation of the brake. These are the Passenger Alarm Circuit shown in figure 3, which detects operation of the Passenger Emergency Buttons (PAB). The Full Speed and Slow Speed Safety Circuits shown in figure 4, which monitor the condition of all the safety devices on the train, and lastly the service braking control wires, which allow the driver to control the brake by means of his Brake Controller. The loss of current in any of the brake circuits will cause an emergency brake application, and special over-ride arrangements are provided to enable the train to be moved under these circumstances.

3.1. Modelling of the Brake System

The failure of the system for which a fault tree is required is the failure of the emergency brake system on demand. The whole braking system has been split into the four sub-systems shown in figures 2, 3, 4, and 5.

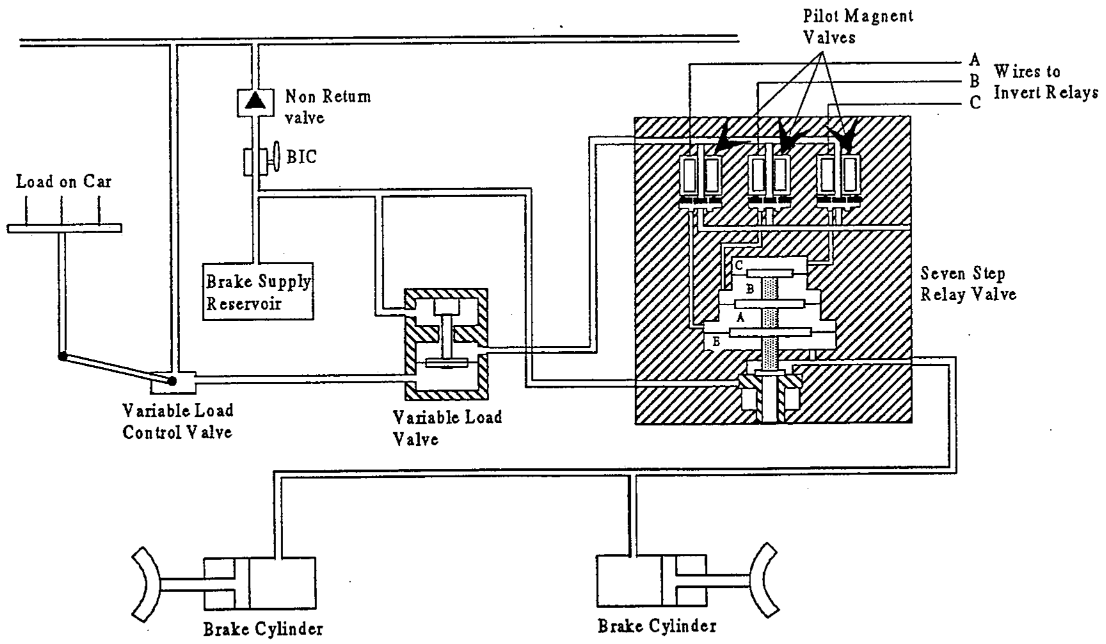

Figure 2 Braking System

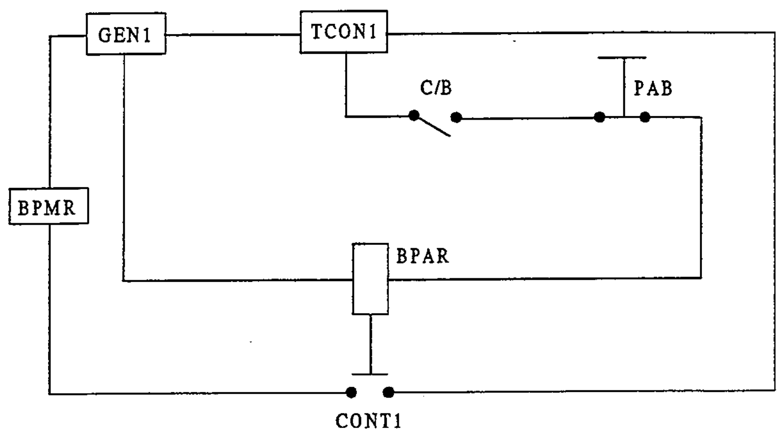

Figure 3 Passenger Alarm Circuit

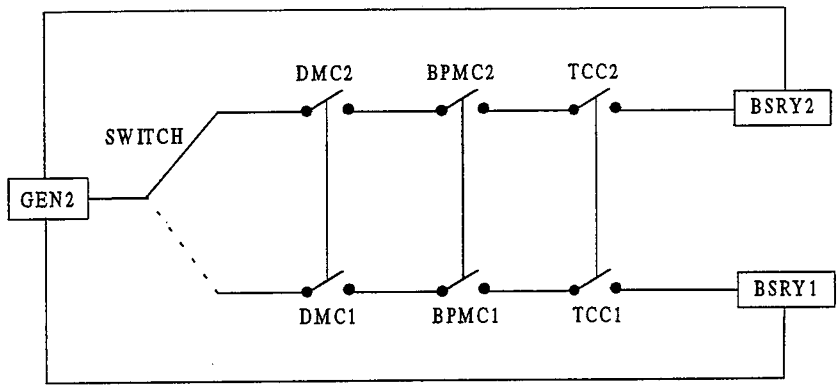

Figure 4 Full and Slow Speed Safety Circuits

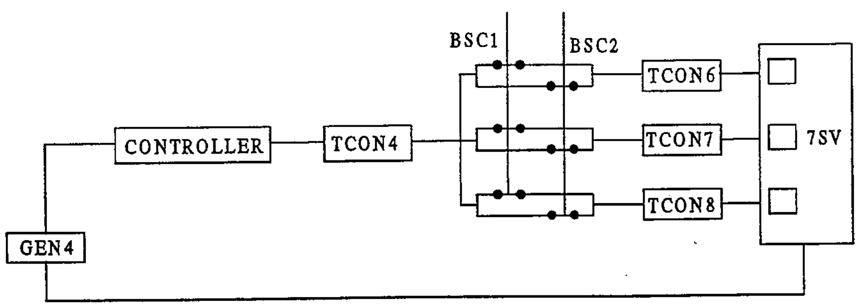

Figure 5 Service Circuits

Each of the four sub-systems are connected. The relay BPMR in figure 3 has its contacts BPMC1 and BPMC2 in the circuits shown in figure 4. The systems shown in figures 4 and 5 are connected by the relays BSRY1 and BSRY2 with the contacts

BSC1 and BSC2 respectively. The service Circuit (figure 5) controls the Seven Step Valve. A listing of the main components in each system follows :

System 1	Generator1 (GEN1)
(fig 3)	Circuit Breaker (C/B)
	Passenger Alarm Button (PAB)
	Brake Passenger Alarm Relay (BPAR)
	Brake Passenger Alarm Contacts (CONT1)
	Brake Passenger Master Relay (BPMR)
System 2	Generator 2 (GEN2)
(fig 4)	Switch
	Deadmans Contacts Full (DMC2)
	Deadmans Contacts SLOW (DMC1)
	Brake Passenger Master Contacts Full (BPMC2)
	Brake Passenger Master Contacts Slow(BPMC1)
	Tripcock Contacts Full (TCC2)
	Tripcock Contacts Slow (TCC1)
	Brake Safety Relay Full (BSR2)
	Brake Safety Relay Slow (BSR1)
System 3	Generator 4 (GEN4)
(fig 5)	Controller
	Brake Safety Contacts Full (BSC2)
	Brake Safety Contacts Slow (BSC1)
	Seven Step Valve
System 4	Compressor
(fig 2)	Non Return Valve (Valve 4)
	Brake Isolating Cock (BIC)
	Brake Reservoir
	Variable Load Valve (Valve 2)
	Variable Load Control Valve (Valve 3)
	Seven Step Valve (Valve 1)
	Pipe
	Brake Cylinder
	Brake Pads

For the system to be modelled the two input files which describe the system had to be constructed. The files are the topology file and the library file which are used by the fault tree generation code.

4. Component models

As an example of how the decision tables are constructed to represent the failure/function of each component, consider the relay contacts (CONT1) shown in figure 3. First the failure modes of the component are identified, in this case the two failure modes are FO (failed open) and FC (failed closed). From figure 3 it can be seen that CONT1 has two inputs and one output, the inputs come from the components TCON1, which is prior to CONT1 in the circuit and the relay controlling the contacts, BPAR and the output is connected to BPMR, the next component in the circuit. In this example the input and output variables only take two states, in general there can be any number of different states for a variable. The variables used are: input 1 C (current), input 2 M (energised) and output 1 C (current). The construction of the decision table can now take place. Each of the different variable states for the output are considered one at a time and a list of their causes determined. The two states of the output variable are Current (C) and No Current (NC). Considering the first of these events Current at output 1 we get the following possible causes:

1. C in IN1 (input 1) , the BPAR relay energised (M in IN2) and the CONT1 working (W in state column).

2. If there is C in IN1 and CONT1 are failed closed (FC) then there will be C in OUT1 (output 1) regardless of IN2.

Turning to the causes for No Current (NC) in output 1, then there are three ways which this result can occur:

1. NC in IN1 will cause NC in OUT1 regardless of IN2 or the state of the component.

2. CONT1 failed open (FO) will cause NC in OUT1 regardless of IN1 and IN2.

3. BPAR de-energised (NM IN2) and the component working will cause NC in OUT1 regardless of IN1.

The completed decision table which represents the causes of each output event in terms of the input events and component states for CONT1 is shown in table 1.

IN1C	IN2M	STATE	OUT1C
C	M	W	C
C	–	FC	C
NC	–	–	NC
–	NM	W	NC
–	–	FO	NC

Table 1 Decision table for the component CONT1

The '–' in the input and state columns indicate "don't care" states.

Another attribute to the decision table is the gain. Gains are entered as '+' or '-' to show how changes in input parameters affect the output parameters. Where an increase in the input parameter increases the output a '+' is entered for the gain. A '-' gain is entered when the output parameter decreases when the input parameter increases.

The full format of a Decision Table is shown in table 3. The table contains:

General component type.
Specific type.
Number of inputs, number of outputs, number of parts.
Gains associated with the relationship between input and output variables.
The headers for the main part of the decision table with their number and variable.
The decision table.
Exclusive row.
Working state and all failures.

The entries in row 4 of the decision table are the gains that are associated with the relationship between the input and output variables.

5. Circuit Identification

The circuits are identified from a topology tree that is formed by the program prior to the start of fault tree construction. The program builds up the topology tree by using information contained in the topology file and the decision table library to link each of the system components as the variable states are traced from the top event back through the system. A topology graph is a directed graph consisting of nodes and edges. The nodes represent specific values of specific variables for the component identified. The edges link the nodes such that those nodes directly below any other nodes contribute to their cause. The topology graph unlike a fault tree does not however indicate how these events combine. As each new node is added to the graph the output column number, input column number and variables are extracted from the decision table and included on the diagram. Termination of each branch on the topology graph occurs when a repeated event or the system boundary are encountered. When the topology graph has been completed all circuits need to be detected and classified.

To identify a circuit from the topology graph a path must exist that starts and ends with the same component with a resulting positive product of the gains.

The relationship between the input variables and output variables for all the components on the loop are extracted from the gain row in the appropriate decision table. The individual gains on the possible loops are multiplied together to give the resultant gain. For the loop to be a circuit the resultant gain must be positive, the variable that is traced around the loop has to be a state of current and the circuit has to contain a power supply. In the decision tables if the component is a power source a "P" is indicated in the row that contains the number of inputs and the number of outputs. If no power source appears in the loop or the variable is not a state of current then the loop is not a circuit.

6. Circuit Operators

Two circuit operators are described below. The first is applicable when the No Current situation is traced from a component on a circuit. This first operator is illustrated in figure 6 and can be summarised as follows:

1. Certain failure modes of any component in the circuit will cause the open circuit condition resulting in No Current. Therefore each component on the circuit is dealt with in turn adding its relevant failure states as inputs to an OR gate.

2. When each component is being dealt with the decision tables are checked to see if there are any external inputs to the circuit at that component. This requires any entries which cause output NC and also have NC in the input that traces the path around the circuit to be identified. The input NC condition requires no further consideration as this just represents the circuit continuity. Other events on the decision table row are developed as external events. If an external input is any state of the variable current then this event is inconsistent with the NC circuit output and can be ignored. Otherwise the external input is identified for further expansion after the operator has been applied.

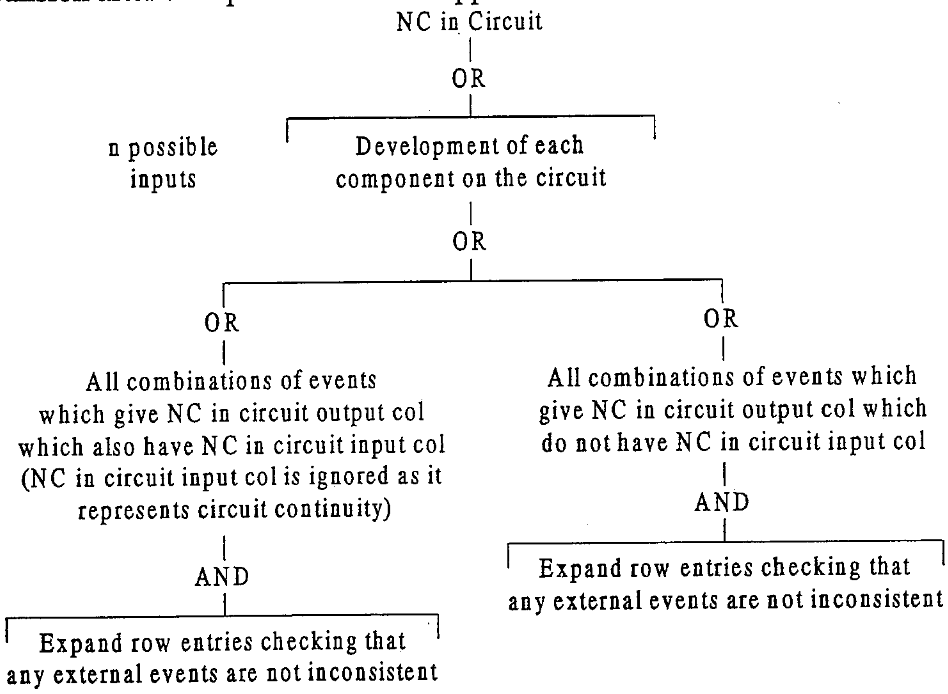

Figure 6 NC in circuit operator

When an event representing Current flowing in a circuit needs development then the second loop operator shown in figure 7 is applied. For this condition all the components in the circuit must be working, so an AND gate appears at the top of the operator.

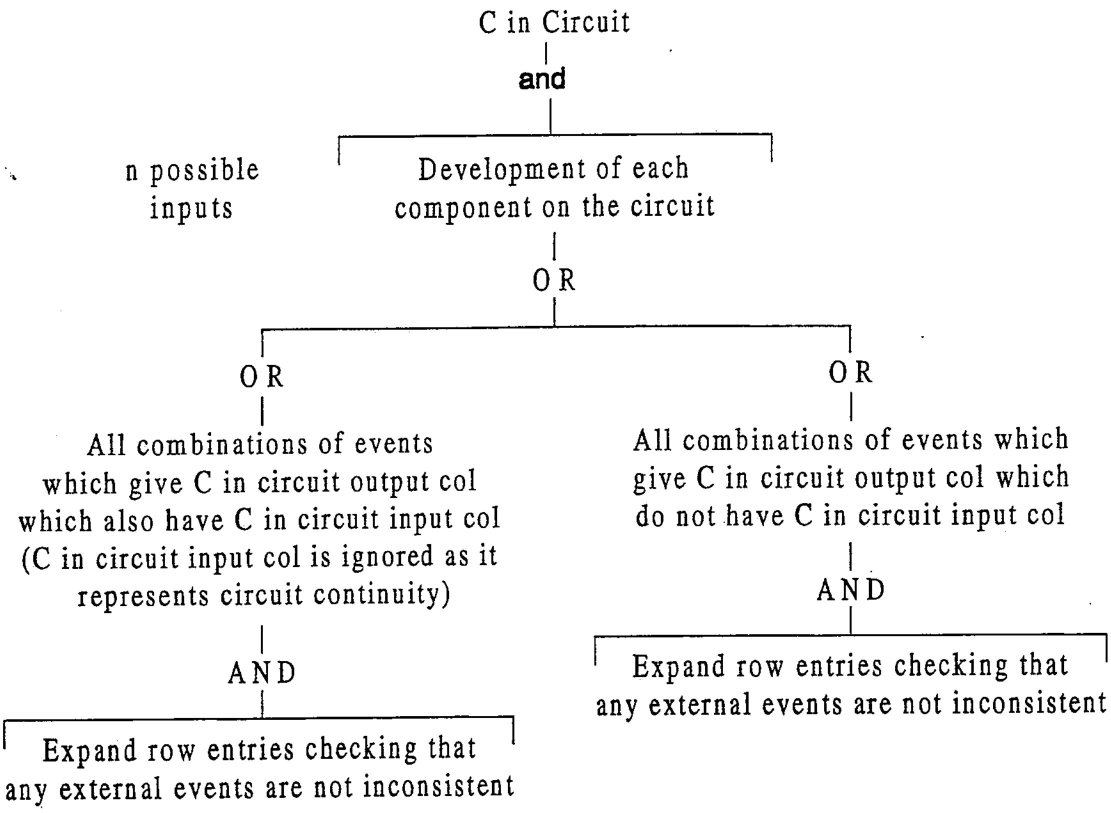

Figure 7 C in circuit operator

Any component working states which appear in the development of the operators are expressed as NOT any of the component failure states. This is necessary to ensure the correct Boolean reduction during the analysis stage.

7. Application to Train Braking System

7.1. Failure of Braking System on Demand

In this analysis the top event of interest is "failure of brakes on one car in an emergency". AFTCC prompts the user for the information specifying the topology file, the file containing the decision tables and the top event. When the program is run for the top event "brakes fail to apply in an emergency", there ten circuits detected. Components appearing on the ten circuits are listed in table 2.

1. Tconnector6 O1 Valve1 O2 Generator4 O1 Controller O1
   Tconnector4 O1 Bsct1 O1

2. Tconnector4 O1 Bsct1 O2 Tconnector7 O1 Valve1 O2
   Generator4 O1 Controller O1

3. Generator2 O1 Switch O2 Dmc2 O1 Bpmc2 O1 Tcc2 O1 Bsry O2

4. Contacts1 O1 Bpmr O3 Generator1 O1 Tconnector1 O1

5. Generator1 O1 Tconnector1 O2 Cirbreak O1 Button O1 Relay2 O1

6. Tcc1 O1 Bsry1 O2 Generator2 O1 Switch O1 Dmc1 O1 Bpmc1

7. Controller O1 Tconnector4 O2 Bsct2 O2 Tconnector7 O1 Valve1 O2 Generator4 O1

8. Tconnector4 O1 Bsct1 O3 Tconnector8 O1 Valve1 O2 Generator4 O1 Controller O1

9. Controller O1 Tconnector4 O2 Bsct2 O3 Tconnector8 O1 Valve1 O2 Generator4 O1

10. Tconnector6 O1 Valve1 O2 Generator4 O1 Controller O1 Tconnector4 O2 Bsct2 O1

Table 2 List of circuits for no emergency brake application

In each of the ten Circuits the variable state of concern is Current (C). The ten circuits appear on three of the four sub-systems of the Brake System. Circuits 4 and 5 come from the Passenger Alarm Circuit shown in figure 3 , Circuits 3 and 6 come from Full and Slow Speed Safety Circuit shown in figure 4 and the remaining Circuits come from the Service Circuit shown in figure 5. Having identified all circuits the program then commences to construct the fault tree for the top event.

As an example of how the decision table method is used to construct a fault tree, the method will be demonstrated by working through the construction of the top three gates for the fault tree in figure 8. First the decision table for the component whose output contains the top event variable is located in the library file in this case it is the decision table for the brake pad (BPAD). The output column is then searched for the occurrence of the top event variable in the specified state, i.e. NP (no pressure). From the decision table for BPAD shown in table 3 there are two matches with the variable NP in output column 1.

BPAD
NORMAL
1, 1, 1
+

I1P	STATE	O1P
HP	W	HP
LP	W	LP
-	SOFF	NP
-	SON	P
NP	W	NP

EXCLUSIVE
W, SON, SOFF

Decision Table 3

As there are two matches an "OR" gate G1 is the first gate introduced in the fault tree construction. Each of the match rows are now dealt with in turn adding inputs to G1. For the second matched row there is more than one entry, namely NP and W, therefore an "AND" gate G2 is added as input to G1 with these two events as inputs. The first matched row is now dealt with, this row only contains one entry, SOFF in the state column. This is a failure of the component (brake pads stuck off) and is added to G1 as the basic event BDSOFF. The inputs to G2 are now developed. From the topology file the component with an input to BPAD is found to be the brake cylinder (BCYL). Causes of NP in the input to BPAD will therefore be NP in output column 1 of the BCYL decision table. These events which give this output are developed as for the component BPAD. The other input to G2 to be developed is W (working).

The "EXCLUSIVE" row holds the working and failure states of the component. The first entry in this row is always the working state. When developing the BPAD working state (W) input to G2 failure states are used instead of adding a working state to the fault tree. This enables the correct Boolean reduction during the analysis stage. As the component has two failure modes an AND gate G3 is added with NOT each failure mode as inputs. The rest of the fault tree is developed in the same manner until an event is encountered that is part of a circuit. Fault tree construction using decision tables is described in more detail in reference 14.

The three events that have yet to be expanded in the fault tree in figure 8 are Tcon6 Col0 (Tconnector 6 output column 0), Tcon7 Col0 and Tcon8 Col0 on the right-hand centre branch. These events are the first occurrences where the circuit operator needs to be applied.

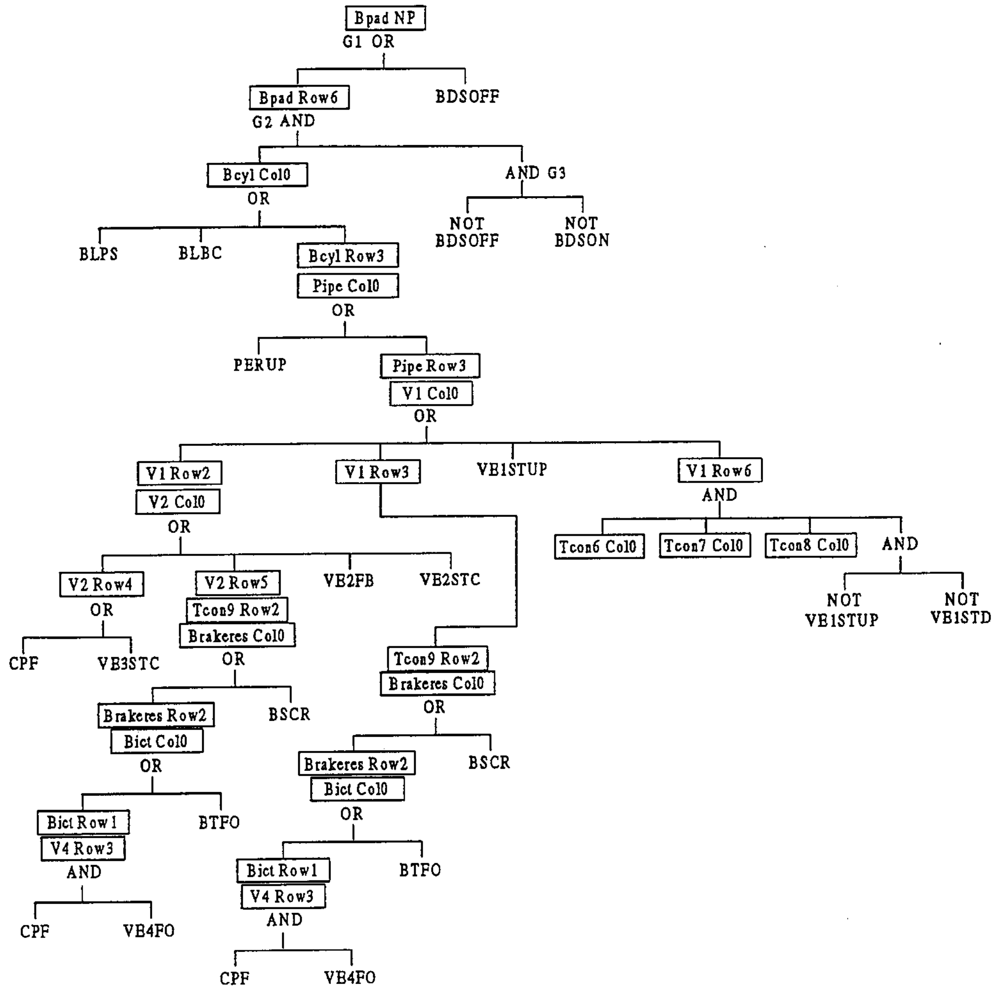

Figure 8 Fault tree for the top event "failure of brakes in an emergency"

The circuits are checked for a match with Tcon6 Col0 C, a match was found in circuits 1 and 10, therefore the entry component (Tcon6) is common to two circuits. There are five common components in the two circuits. The common components in circuits 1 and 10 are Tconnector6, Valve1, Generator4, Controller and Tconnector4.

CONTROLLER
NORMAL
1, 1, 1
+, +

I1C	STATE	O1C
C	W	C
NC	-	NC
-	F	NC

EXCLUSIVE
W, F

Decision Table 4

TCONNECTOR
CONTT
2, 1, 1
+, +

I1C	I2C	STATE	O1C
C	-	-	C
-	C	-	C
NC	NC	-	NC

EXCLUSIVE
W, F

Decision Table 6

GENERATOR
1IN1OUT
1, 1, 1, P
+

I1C	STATE	O1C
C	W	C
NC	-	NC
-	F	NC

EXCLUSIVE
W, F

Decision Table 5

TCONNECTOR
1IN2OUT
1, 2, 1
+, +

I1C	STATE	O1C	O2C
C	-	C	C
NC	-	NC	NC

EXCLUSIVE
W, F

Decision Table 7

BSCT
NORMAL
2, 3, 1
+, +, +, +, +,+

I1C	I2M	STATE	O1C	O2C	O3C
C	M	W	C	C	C
NC	-	-	NC	NC	NC
-	-	FO	NC	NC	NC
C	-	FC	C	C	C
-	NM	W	NC	NC	NC

EXCLUSIVE
W, FO, FC

Decision Table 8

VALVE
7SV
5, 2, 1
+, +, +, +, +, +, +, +, +, +

I1P	I2P	I3C	I4C	I5C	STATE	O1P	O2C
HP	P	NC	NC	NC	W	HP	NC
NP	-	-	-	-	-	NP	-
-	NP	-	-	-	-	NP	-
-	P	-	-	-	STD	P	-
-	-	-	-	-	STUP	NP	-
-	-	C	C	C	W	NP	C

EXCLUSIVE
W, STUP, STD

Decision Table 9

Decision tables for the components on the two circuits are shown in Decision Tables 4-9.

The general circuit operator for NC for nested circuits with a common entry component is shown in figure 9

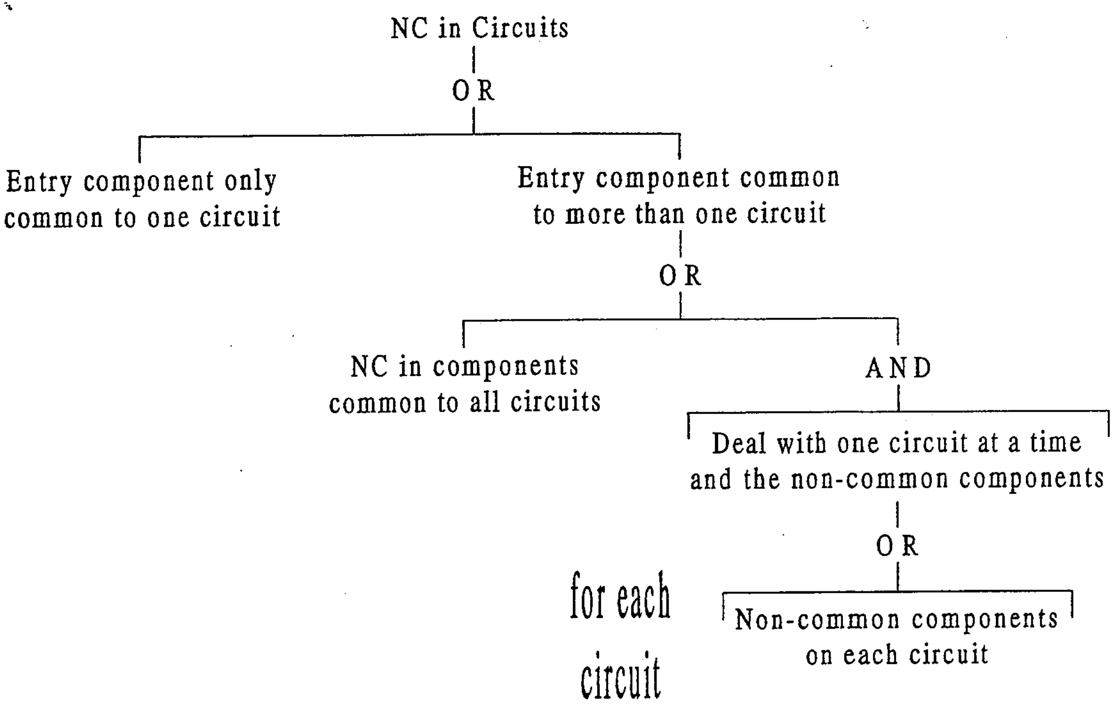

Figure 9 General circuit operator for NC

The circuit operator for NC in figure 6 is applied at the end of each branch of the general operator for NC figure 9. The general operator for C is very similar and is shown in figure 10

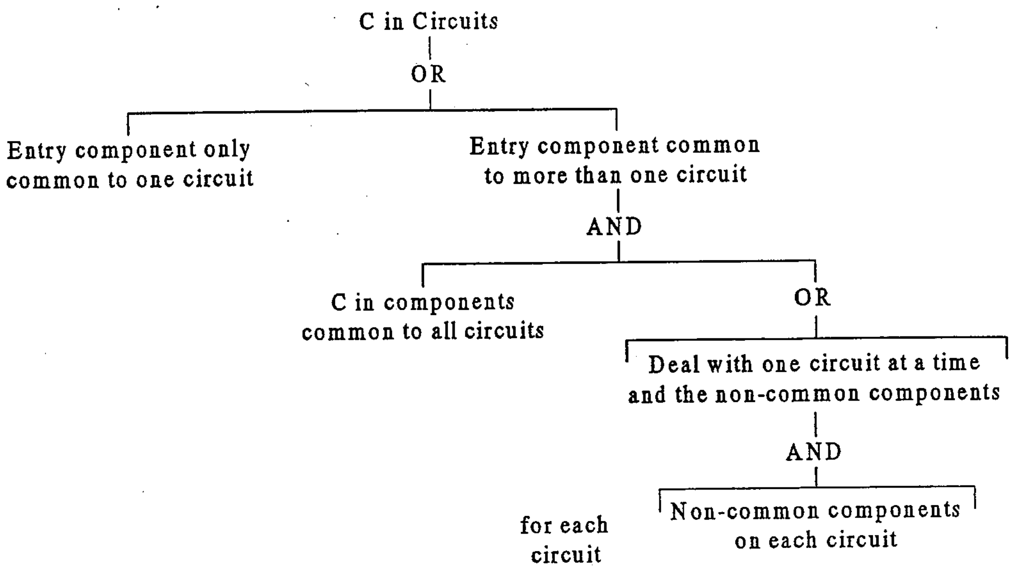

Figure 10 General circuit operator for C

The circuit operator is applied and the common components are dealt with first. As the variable state being traced is current (C) the circuit operator for current, figure 10, is used. Since Tcon6 is common to two circuits the first gate to be added is an AND, with inputs from the common components. The decision table 6 for the component Tconnector6 is found and output column 1 is searched for the variable state current, there are two matches in transition rows 1 and 2. The matches give C, -,- and -,C,- as the causes so only these two entries that need to be considered. For the first of these entries the first event is current in input 1 of Tconnector6 this input comes from the Brake Safety Contacts Slow (Bsct1) output 1. All of the circuits are checked to see if Bsct1 O1 appears. It only appears on circuit 1 which is one of the circuits that the operator is currently being applied to, so this entry is just tracing the current round the circuit and since the circuit is considered as a whole by the operator it needs no further consideration. The next input, row 2 -, C, - is dealt with in the same manner, this time the input event current is traced to the component Brake Safety Contacts Full output column 1 (Bsct2 O1) which only appears on circuit 10 which is the second circuit that is being dealt with and therefore again traces the circuit continuity and is not considered further. So for the common component, Tconnector6, no entries are added to the AND gate. The next common component in the circuit lists is Valve1 O2, decision table 9. There is only one match found for the event current in output column 2 of the decision table, which is row 6. Before any events are examined the descriptor Valve 1 row 6 is checked to see if it is a repeated event, in this case it is (it occurs as the output of the gate in figure 8 which is currently being developed) and therefore can be disregarded. The rest of the common components are dealt with as described for the Tconnector and Valve to produce the fault tree structure in figure 11.

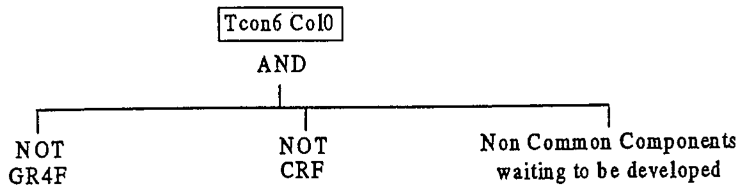

Figure 11 Application of the Circuit Operator to Tconnector6 Col0

The branch waiting to be expanded in figure 11 is the branch that will contain the components that are on one circuit but are not common to both of the two circuits. The two non-common components are Bsct1 O1 (Brake Safety Contacts Slow) on circuit 1 and Bsct2 O1 (Brake Safety Contacts Full) on circuit 10. To produce the correct structure of the tree all combinations of the failures of the non-common components on each circuit must be taken together. This results in an OR gate added as input to the initial AND gate on the circuit operator which has as its inputs failures occurring on the non-common sections of each circuit. This results in the fault tree shown in figure 12

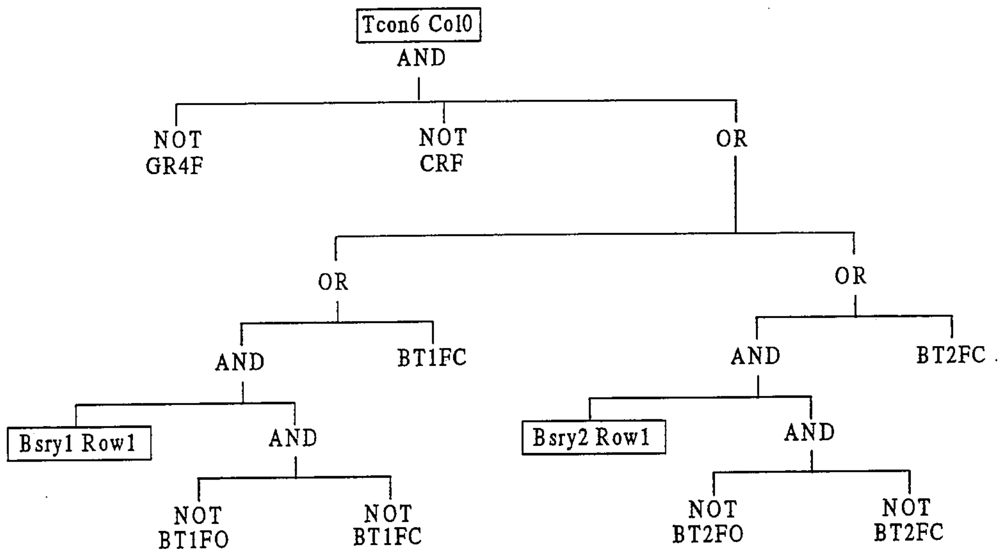

Figure 12 Completed Application of the Circuit Operator for Multiple Circuits

The rest of the tree is expanded applying the circuit operator whenever a component is encountered which lies on one or more of the circuits. When the program has terminated the fault tree structure is contained in a data file in the format for direct input into a fault tree analysis code.

The causes of an emergency brake application are: release of the deadmans handle, activation of the passenger alarm circuit and activation of the tripcock. Therefore three separate fault trees need to be constructed for failure of the brake system when it is activated by each of these causes when a demand occurs. To model each situation the top event of the fault tree developed by AFTCC is ANDED with the cause of the emergency brake application demand.

Having applied the checks for repeated and inconsistent events in the fault tree structure the resulting fault trees are analysed. With the working states removed the prime implicant sets agreed with the minimal cut sets produced by a manual analysis.

7.2. Fault Tree Development for Spurious Brake Applications

The second analysis for the system developed causes of spurious emergency brake application. This event is of particularly concern as stationary trains can also pose a major hazard.

The system is described by the same topology file and decision tables. Development of the fault tree for spurious brake application only requires the program input parameter which specified the top event to be changed.

The automatic construction of the tree is performed in a similar manner to the previous example, however in this case the variable state in the first circuit that is encountered during development is NC (no current) so the operator shown in figure 10 is applied. Analysis of the resulting fault tree produces 44 minimal cut sets. These cut sets again agree with the cut sets calculated produced by a manual construction of the tree.

8. Conclusion

The automatic fault tree construction package has been tested on a train braking system which incorporated multiple application of the new developed circuit operators. The use of the circuit operators produces a more efficient algorithm reducing the computational time required to test the logical consistency of repeated events. It also has the advantage of significantly reducing the size of constructed fault trees. Upon construction of a fault tree the program creates a data file that holds the fault tree information in the correct format for direct input into a fault tree analysis package. The structure of the trees produced by applying the circuit operator are also of a more "minimal" form which represents the system failure logic in a more efficient way.

9. References

1. Andrews, J.D. and Moss, T.R., Reliability and Risk Assessment, Longmans, 1993.

2. USAEC 1974, An Assessment of Accidental Risk in US Commercial Nuclear Power Plants, WASH 1400, August 1974.

3. Lapp, S.A. and Powers, G.J., Computer-aided Synthesis of Fault Trees, IEEE Trans Reliability, R–26 (April 1977), 2–13.

4. Andrews, J.D. and Morgan, J.M., Application of the Digraph Method of Fault Tree Construction to Process Plant, Reliability Engineering, 14/27, 1986, 85–106.

5. Andrews, J.D. and Brennan, G., Application of the Digraph Method of Fault Tree Construction to a Complex Control Configuration, Reliability Engineering and System Safety, 28, 1990, 357–384.

6. Salem, S.L., Apostalarkis, G.E. and Okrent, A Computer-oriented Approach to Fault Tree Construction, EPRI Report NP–288, Electric Power Research Institute, 1976.

7. Xie, G., Xue, D. and Xi, S. Tree-Expert: a tree based expert system for Fault Tree Construction, Reliability Engineering and System Safety, vol 40, 1993, 295–309.

8. Hunt, A., Kelly, B.E., Mullhi, J.S., Lees, F.P. and Rushton, A.G., The propagation of faults in process plants 6, Overview of, and modelling for, fault tree synthesis, Reliability Engineering and System Safety, vol 39, 1993, pp 173–194.

9. Hunt, A., Kelly, B.E., Mullhi, J.S., Lees, F.P. and Rushton, A.G., The propagation of faults in process plants: 7, Divider and header units in fault tree synthesis, Reliability Engineering and System Safety, vol 39, 1993, pp 195–209.

10. Hunt, A., Kelly, B.E., Mullhi, J.S., Lees, F.P. and Rushton, A.G., The propagation of faults in process plants: 8, Control systems in fault tree synthesis, Reliability Engineering and System Safety, vol 39, 1993, pp 211–227.

11. Hunt, A., Kelly, B.E., Mullhi, J.S., Lees, F.P. and Rushton, A.G., The propagation of faults in process plants: 9, Trip systems in fault tree synthesis, Reliability Engineering and System Safety, vol 39, pp 229–241.

12. Hunt, A., Kelly, B.E., Mullhi, J.S., Lees, F.P. and Rushton, A.G., The propagation of faults in process plants: 10, Fault tree synthesis-2, Reliability Engineering and System Safety, vol 39, 1993, pp 243–250.

13. Andow, P.K., Difficulties in fault-tree synthesis for process plant. IEEE Trans Reliability, R-29(1) (April 1980) 2–9.

14. Henry, J.J. and Andrews, J.D. An Approach to Fault Tree Synthesis for Railway Safety Systems. Proceedings of ESREL 1995, vol 2, pp 623–640.