

THE UNIVERSITY OF HULL

Quantitative Analysis of Dynamic Safety-Critical Systems
Using Temporal Fault Trees

being a Thesis submitted for the Degree of
Doctor of Philosophy
in the University of Hull

by

Ernest Edem Edifor

August 2014

**QUANTITATIVE ANALYSIS OF DYNAMIC SAFETY-CRITICAL SYSTEMS
USING TEMPORAL FAULT TREES**

ABSTRACT

Emerging technological systems present complexities that pose new risks and hazards. Some of these systems, called safety-critical systems, can have very disastrous effects on human life and the environment if they fail. For this reason, such systems may feature multiple modes of operation, which may make use of redundant components, parallel architectures, and the ability to fall back to a degraded state of operation without failing completely. However, the introduction of such features poses new challenges for systems analysts, who need to understand how such systems behave and estimate how reliable and safe they really are.

Fault Trees Analysis (FTA) is a technique widely accepted and employed for analysing the reliability of safety-critical systems. With FTA, analysts can perform both qualitative and quantitative analyses on safety-critical systems. Unfortunately, traditional FTA is unable to efficiently capture some of the dynamic features of modern systems. This problem is not new; various efforts have been made to develop techniques to solve it. Pandora is one such technique to enhance FTA. It uses new 'temporal' logic gates, in addition to some existing ones, to model dynamic sequences of events and eventually produce combinations of basic events necessary and sufficient to cause a system failure. Until now, Pandora was not able to quantitatively evaluate the probability of a system failure. This is the motivation for this thesis.

This thesis proposes and evaluates various techniques for the probabilistic evaluation of the temporal gates in Pandora, enabling quantitative temporal fault tree analysis. It also introduces a new logical gate called the 'parameterised Simultaneous-AND' (pSAND) gate. The proposed techniques include both analytical and simulation-based approaches. The analytical solution supports only component failures with exponential distribution whilst the simulation approach is not restricted to any specific component failure distribution. Other techniques for evaluating higher order component combinations, which are results of the propagation of individual gates towards a system failure, have also been formulated. These mathematical expressions for the evaluation of individual gates and combinations of components have enabled the evaluation of a total system failure and importance measures, which are of great interest to system analysts.

ACKNOWLEDGEMENT

I have longed for this opportunity to say thank you to Yahweh for his grace, love, peace that surpasses all understanding, strength and protection that has brought me this far.

I could not ask for a lovelier wife and daughters – Anita, Michelle and Gabriella – who have been very patient with me and have endured my absence for several months. Neither could I ask for a more helpful family – parents, siblings and in-laws – for their support, encouragement and prayers. Thanks to all my family friends – Owusus, Sikankus, Watkins and Frazers – for their companionship.

Very many people endure their PhDs but I have thoroughly enjoyed mine and I owe so much to Dr Neil Gordon who has been an excellent supervisor, support, counsellor and friend. I will also like to say a special thank you to Dr Martin Walker, who has ‘shaped’ and led me through the ‘Pandoran jungle’ of reliability engineering and Prof Yiannis Papadopoulos and Dr Leonardo Bottaci whose constructive criticisms and directions have been instrumental in the production of this thesis.

I will also like to thank all the staff members in the computer science department, especially Dr David Parker, Dr Darryl Davis, Dr Chandra Kambhampati, Amanda Milson, Mark Bell, Mike Bielby, David Glover, Simon Grey, Adam Hird, Andrew Hancock, Helen El-Sharkawy, Lynn Morrel, Jo Clappison, Sally Byford and Colleen Nicholson (who is no longer in the department), for their academic, administrative and technical support. I will also like to express my sincere gratitude to the cleaners of the department, especially Sue Gibson who did not only make the PhD office a better place to work but also became a personal friend to have brief early morning chats with.

To all my fellow colleagues, who made the PhD experience more bearable, by saving me from getting lost in the “Pandoran Jungle” through engagements in sports and other social events, I say thank you – Zhibao Mian, Luis Azevedo, Sohag Kabir, John Dixon, Mustafizur Rahman, Lamis Farah Al-Qora’n, Lisa Moore, Poolsawad Nongnuch, Nabil Abu Hashish, Julius Nganji, Dr Shawulu Nggada, Dr Amer Dhadeen and Dr Septavera Sharvia.

I will like to say a big thank you to all my friends and relations in Hull and abroad who have helped me and my family in various ways: Rev. Clifford Kasim and the Initiators of Change Ministries, Pastor Isaac Aleshinloye and the Amazing Grace Chapel.

The entire PhD research was fully supported by the following bodies: University of Hull, Widening Participation, MAENAD (FP7 Grant 260057) - Modelling Analysis Evaluation of Novel Architectures for Dependable Electric Vehicles and European Research Project (with Fiat, Volvo, 4S, Continental, Delphi, French Atomic Authority, TU Berlin, and RIT Stockholm). I am very grateful to all these bodies.

Finally, to you the reader of this thesis, I say thank you for considering it worthy to be read.

TABLE OF CONTENTS

ABSTRACT.....	III
ACKNOWLEDGEMENT	IV
TABLE OF CONTENTS.....	VI
NOTATION.....	X
LIST OF ABBREVIATIONS.....	XI
LIST OF FIGURES	XIII
LIST OF TABLES.....	XV
Chapter One	1
INTRODUCTION	1
1.1 Field of Research.....	1
1.2 Motivation and Scope	4
1.3 Research Question.....	5
1.4 Research Contribution.....	6
1.5 Assumptions.....	8
1.6 Structure	9
1.7 Note on Publication.....	11
Chapter Two	12
BACKGROUND	12
2.1 Reliability Engineering	12
2.1.1 Reliability Theory	14
2.1.2 Probability Theory	16
2.2 Fault Tree Analysis (FTA).....	22
2.2.1 Qualitative Analysis.....	27
2.2.2 Quantitative Analysis.....	29
2.2.3 Limitations of FTA	41

2.3	Dynamic Fault Trees (DFT).....	44
2.3.1	DFT Qualitative Analysis	47
2.3.2	DFT Quantitative Analysis	48
2.3.3	Limitations of DFT	51
2.4	Temporal Fault Tree (TFT) Analysis.....	52
2.4.1	Qualitative Analysis.....	55
2.4.2	TFT Quantitative Analysis.....	57
2.4.3	Limitations of TFT.....	57
2.5	Review of Quantitative Techniques	58
Chapter Three		61
TEMPORAL QUANTITATIVE ANALYSIS		61
3.1	SAND Gate Quantification	61
3.1.1	Behavioural and Timing Models	61
3.1.2	Analytical Model	62
3.2	Non-Inclusive PAND Gate Quantification	66
3.2.1	Behavioural and Timing Models	66
3.2.2	Analytical Model	67
3.2.3	Monte Carlo Solution.....	68
3.3	POR Gate Quantification	69
3.3.1	Behavioural and Timing Models	69
3.3.2	Analytical Model	70
3.3.3	Monte Carlo Solution.....	79
3.4	Parameterised-SAND Gate Quantification	80
3.4.1	Behavioural and Timing Models	82
3.4.2	Analytical Model	83
3.4.3	Monte Carlo Solution.....	86
3.5	MCSQ and Top-Event Quantification	86

3.5.1	MCSQ with One Type of Operator.....	87
3.5.2	MCSQ with Different Operators.....	88
3.6	Evaluating Systems with Different Failure Distributions	91
Chapter Four	96
TEMPORAL QUALITATIVE ANALYSIS	96
4.1	Groups and Modules Modularization Technique (GMMT).....	96
4.1.1	Classification	97
4.1.2	Combination.....	99
4.1.3	Comparison.....	101
4.1.4	Strengths and Limitations of the GMMT Approach.....	101
4.2	Temporal Binary Decision Diagrams (TBDD)	102
4.2.1	Introducing Temporal Binary Decision Diagrams.....	103
4.2.2	Case Study: Hypothetical System.....	109
4.2.3	Strengths and Limitations TBDD Approach	115
Chapter Five.....	116
CASE STUDY.....	116
5.1	Aircraft Fuel System (AFS)	116
5.1.1	Operational Behaviour of AFS	119
5.1.2	Temporal Fault Tree Analysis	120
5.2	Qualitative Analysis of AFS	123
5.3	Quantitative Analysis of AFS	127
Chapter Six	137
EVALUATION	137
6.1	Evaluation of Techniques.....	137
6.1.1	Evaluation of Quantitative Techniques.....	137
6.1.2	Evaluation of Qualitative Techniques.....	138
6.2	Evaluation Against Objectives.....	140

6.3	Limitations and Further Research	144
Chapter	Seven	149
CONCLUSION	149
REFERENCES	153
APPENDIXES	167
APPENDIX 1: Monte Carlo Condition for Various MCSQs	167
APPENDIX 2: Algorithm to Evaluate Top-Event.....		169
APPENDIX 3: Preliminary List of CSQs.....		171
APPENDIX 4: Supply Chain Risk Management Using Pandora.....		178
GLOSSARY	184

NOTATION

$f(S)\{t\}$	PDF of a system S and a mission time t
$F(S)\{t\}$	CDF of a system S and a mission time t
\emptyset	Empty set
$ePAND$	Exclusive PAND
λ_i	Failure rate of event i
$\lambda(S)\{t\}$	Failure rate of a system S and a mission time t
$h(S)\{t\}$	Hazard function of a system S and a mission time t
$iPAND$	Inclusive PAND
\cap	Intersection in set theory
\bullet	Logical AND
\neg	Logical NOT
$+$	Logical OR
μ_N	Mean of normal distribution
t	Mission time or system lifetime
\triangleleft	Non-Inclusive Before
$\&_d$	parameterized Simultaneous-AND with a duration of interval d
$<$	Priority-AND
$ $	Priority-OR
$P(S)\{t\}$	Probability of a system S failing within a mission time t
$Q(S)\{t\}$	Unreliability of a system S and a mission time t
$R(S)\{t\}$	Reliability of a system S and a mission time t
α	Scale parameter of Weibull distribution
β	Shape parameter of Weibull distribution
$\&$	Simultaneous-AND
σ_N	Standard deviation of normal distribution
Φ	Standard normal CDF for lognormal distribution
ϕ	Standard normal PDF for lognormal distribution
$TTF(S)\{t\}$	Time-to-failure of a system S and a mission time t
Ω	Universal set

LIST OF ABBREVIATIONS

BDD	Binary Decision Diagram
BM	Birnbaum Importance Measure
CCF	Common Cause Failure
CDF	Cumulative Distribution Function
CSP	Cold-Spare
CTMC	Continuous Time Markov Chains
DFT	Dynamic Fault Tree
DNF	Disjunctive Normal Form
EP	Esary-Proschan
ePAND	Exclusive Priority-AND
FDEP	Functional Dependency
FMEA	Fault Modes Effects Analysis
FTA	Fault Tree Analysis
FV	Fussell-Vesely
GMT	Groups and Modules Table
GMMT	Groups and Modules Modularization Technique
HiP-HOPS	Hierarchically Performed Hazard Origin Propagation Studies
HPS	Hospital Power System
iPAND	Inclusive Priority-AND
LTA	Linear Time Algorithm
MC	Monte Carlo
MCS	Minimal Cut Set
MCSQ	Minimal Cut Sequence
MTTF	Mean Time-to-Failure
PAND	Priority-AND
PDF	Probability Density Function
PIE	Principle of Inclusion-Exclusion
POR	Priority-OR
PRA	Probabilistic Risk Assessment
PSA	Probabilistic Safety Assessment
RAW	Risk Achievement Worth

RBD	Reliability Block Diagram
RPN	Reverse Polish Notation
RRW	Risk Reduction Worth
SAND	Simultaneous-AND
SFTA	Static Fault Tree Analysis
TBDD	Temporal Binary Decision Diagram
TFT	Temporal Fault Tree
TTF	Time-to-Failure

LIST OF FIGURES

Figure 1.1-1: A 2-Fuel Hospital Power Supply (HPS) system	3
Figure 2.1-1: Failure rate curve	15
Figure 2.2-1: Graphical representations of the AND gate	24
Figure 2.2-2: Timing diagram of the AND gate with 2 input event	24
Figure 2.2-3: Graphical representations of the OR gate	25
Figure 2.2-4: Timing diagram of the OR gate with 2 input event	25
Figure 2.2-5: An abstract model of HPS	26
Figure 2.2-6: Graphical representation of an event X	29
Figure 2.2-7: Timing diagram of an event X	29
Figure 2.2-8: BDD and its structure for a basic event X	32
Figure 2.2-9: BDD for HPS using $MD \rightarrow E \rightarrow I \rightarrow G \rightarrow S$	34
Figure 2.2-10: BDD for HPS using $S \rightarrow MD \rightarrow I \rightarrow E \rightarrow G$	36
Figure 2.2-11: Graph for Monte Carlo simulation of HPS	39
Figure 2.3-1: A (right): PAND gate for n events, B (left): equivalent SFT	44
Figure 2.3-2: A cold spare gate with n events	45
Figure 2.3-3: A spare component for two primary components	46
Figure 2.3-4: An FDEP gate with a trigger and n dependent events	47
Figure 2.4-1: A PAND gate with n input events	54
Figure 2.4-2: A SAND gate with n input events	54
Figure 2.4-3: A POR gate with n input events	55
Figure 3.1-1: Timing models of a POR gate	70
Figure 3.1-2: Markov model of a POR gate	71
Figure 3.2-1: Timing models of an ePAND gate	67
Figure 3.2-2: Graph of n events occurring at t	67
Figure 3.3-1: Timing models of an SAND gate	62
Figure 3.4-1: pSAND graphical representations	82
Figure 3.4-2: Timing models of a pSAND gate	83
Figure 3.4-3: Graph of two nearly simultaneous events	84
Figure 4.2-1: Basic TBDD irte structure	103
Figure 4.2-2: Directed binary tree	104
Figure 4.2-3: Balanced sequence	104

Figure 4.2-4: Directed binary tree of $A.B$	105
Figure 4.2-5: Hypothetical model	109
Figure 4.2-6: TBDD for $A \rightarrow E \rightarrow B \rightarrow F \rightarrow C \rightarrow D \rightarrow G$	111
Figure 5.1-1: Aircraft Fuel System model	117
Figure 5.1-2: AFS temporal fault tree for “Failure of Starboard Feed”	122
Figure 5.3-1: Monte Carlo estimates for Weibull distribution	135
Figure 5.3-2: Monte Carlo estimates for Log-normal distribution	135
Figure 6.1-1: Breakdown of thesis’ aim	140

LIST OF TABLES

Table 2.1-1: Summary of Exponential Distribution (O'Connor, 2011)	19
Table 2.1-2: Summary of Lognormal Distribution (O'Connor, 2011)	19
Table 2.1-3: Summary of Weibull Distribution (O'Connor, 2011)	20
Table 2.2-1: FTA's Logical Gates	23
Table 2.2-2: Failure Probabilities of HPS' Events	30
Table 2.5-1: Comparison of Quantitative Techniques	59
Table 5.3-1: Component Failure Data	128
Table 5.3-2: pSAND Gate Time Intervals	128
Table 5.3-3: MCSQs Probabilities and Importance for $t = 1$	129
Table 5.3-4: FV Importances of Components $t=1hr (0.006 wks)$	130
Table 5.3-5: BM Importances of Components $t=1hr (0.006 wks)$	130
Table 5.3-6: AFS Top-Event Probabilities using Exponential Distribution	131
Table 5.3-7: AFS Top-Event Probabilities using Weibull Distribution	132
Table 5.3-8: AFS Top-Event Probabilities using Lognormal Distribution	133
Table 5.3-9: Characteristics of Figures 5.3-1 and Figure 5.3-2	136
Table 6.3-1: Effect of dynamic stopping technique 1	146
Table 6.3-2: Effect of dynamic stopping technique 2	146

Chapter One

INTRODUCTION

1.1 Field of Research

More and more, technology is being integrated into every facet of human life, increasing our dependence on it. Though technology is typically intended to open up new possibilities, allow us to work faster and make life easier, the failure of some of these technological systems can be catastrophic, causing harm to both human life and the environment. Such systems are known as **safety-critical systems** or high-consequence systems. Therefore, it is imperative that such systems provide a high level of reliability with low levels of risks and hazards.

For various reasons – such as the development of new systems and industries, the changing nature of accidents, the emergence of new vulnerabilities and exposure to hazards – technological risks keep increasing (Leveson, 2011), making the issue of safety and reliability of systems vital; the importance of this issue cannot be understated. For example, for engineers in the automotive, nuclear and aeronautic industries, the reliability of the systems they design and manufacture is a core requirement. This requirement has given birth to the field of reliability engineering: to analyse systems for their safety (ability to cause no harm) and reliability (ability to perform their intended functions within a specific duration and in specific conditions).

There are several techniques for analysing the reliability of systems. A reliability engineering technique that has served as the genesis of various modern techniques is Fault Tree Analysis (FTA). FTA was created in the 1960s to study the Minuteman Launch Control System (Ericson, 1999). It is a top-down deductive method of analysing the root causes of hazards. It usually commences by identifying a hazard or an undesired event or failure known as the **top-event**. This is followed by the determination of combinations of individual component faults known as **basic event** or input events, which can trigger the occurrence of the top-event. Basic events or combinations of them, known as **intermediate events**, are connected to the top-events with logical Boolean connectives: AND and OR.

FTA can be performed **quantitatively** (probabilistically) or **qualitatively** (logically). Logical analysis of FTA involves using Boolean algebra to obtain a logical expression that contains all the **minimal cut sets** (MCS) resulting in the top event occurrence. MCSs are combinations of logically related events necessary and sufficient to cause the top event: necessary because each basic event in the MCS is needed for the top event to occur and sufficient because the MCS does not need the occurrence of additional events to cause the top event occurrence. This evaluation gives valuable information, such as the critical components – those that possess individual faults that by themselves can lead to the failure of a system – to engineers and system analysts. Based on this information, engineers can redesign their systems to enhance system reliability and make better economic decisions before the system is actually produced.

On the other hand, quantitative analysis usually involves the calculation of how reliable a system is. It involves making use of the probabilities of basic events in evaluating the probabilities of each MCS and the top-event and eventually the relative contribution between MCSs to the occurrence of the top-event.

Even though traditional FTA has achieved many successes, it is handicapped when the sequence of events is considered. To illustrate this short-coming, consider a medical Hospital Power System (HPS) with two main power sources: electricity and generator. It is likely that medical service providers will use dual power sources (especially if the primary source of power is not very reliable) due to the criticality of the services (surgical equipment, incubators and intensive care units) they provide. Assume that the primary source of power is a mainstream electric (E) power supply and the standby power source is a generator (G). Let ' I ' represent an input command, such as a switch, to power a medical device, MD ; and let S represent a sensor that activates the generator system when it detects an omission in the electricity system. An abstract diagram of the entire system is given below in Fig. 1.1-1.

Using the classical FTA, the MCSs for total failure of the medical device are as follows:

1. failure of the medical device (MD) itself
2. failure of the electricity system (E) and generator system (G)
3. failure of the electricity system (E) and sensor (S)
4. omission of the input command (I)

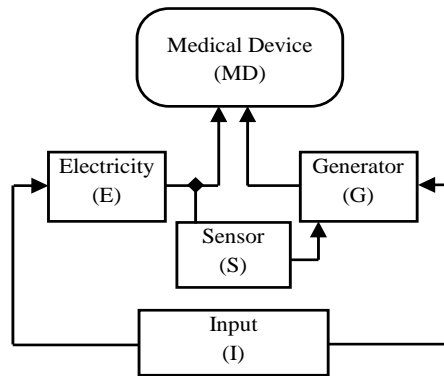


Figure 1.1-1: A 2-Fuel Hospital Power Supply (HPS) system

With this list of MCSs and the individual component failure rates, the probability of *MD* failing can be determined. However, a critical study of these MCSs reveals some inaccuracies when the system is considered to be a real world system where the order in which individual fuel sub-systems fail is essential. For example, what will happen if the sensor fails before the main electricity power supply or vice versa? Traditional FTA is unable to capture, incorporate and analyse these situations in its analysis.

A more precise analysis of HPS, considering the sequential order in which events fail, will be:

1. omission of an input command (*I*)
2. failure of the medical device itself (*MD*)
3. failure of the sensor (*S*) before failure of the main power supply (*E*)
4. simultaneous failure of the sensor (*S*) and the electricity system (*E*)
5. failure of both electricity (*E*) and generator (*G*) systems

Though the results from classical FTA and that from the sequential ordering of events have common causes of failure of *MD*, there are few significant differences. For example, from the above results it is evident that more information about the dynamic behaviour of HPS is not captured by FTA. For example, the third point, ‘failure of the sensor (*S*) before failure of the main power supply (*E*)’ will cause a failure in a medical device because the failure of the electricity system will not be captured by the sensor (which has already failed), and thus the generator system will not get activated and finally no power supply to the medical device. However, failure of the electricity system before the sensor will not necessarily lead to a failure in the medical system because immediately after the electricity system fails, the sensor detects its omission and activates the generator system, which supplies power to the medical device afterwards. Failure of the sensor after this point will not affect the operation of the medical device.

This means that the third point ‘failure of the electricity system (E) and sensor (S)’ from the previous results using classical FTA is inaccurate because it suggests that whenever E and S fail – E fails before S or S fails before E or both E and S fail simultaneously – the medical device will fail. Traditional FTA fails to capture these critical dynamic scenarios in contemporary high-consequence systems. This is a major drawback because it could lead to inaccurate evaluation of both qualitative and quantitative analyses of safety-critical systems.

FTA, since its inception, has seen several modifications to enable it to overcome the limitation of the particular sequence of events. One significant and predominant modification is the Dynamic Fault Tree (DFT) (Dugan et al., 1992; Dugan, 2001). DFTs, with the introduction of new gates and quantitative analysis, addressed (to some extent) the issue of sequential event occurrence deficient in traditional FTA. Another recent significant modification to the classical FT is the Temporal Fault Tree (TFT) (Palshikar, 2002) which specifies temporal dependencies between events and introduces more formal semantics for analysis.

Pandora (Walker, 2009; Walker & Papadopoulos, 2007) is yet another recent modification of FTA which introduces novel gates (Priority-OR and Simultaneous-AND) in addition to existing gates (Priority-AND, Boolean AND and OR) and provides a system of logical laws to quantitatively analyse fault trees by considering the order in which events occur. The end product of Pandora’s qualitative analysis is a list of **Minimal Cut Sequences** (MCSQs), which are sequences of events necessary and sufficient to cause a top event.

1.2 Motivation and Scope

As earlier discussed (and will be seen in detail later), FTA is very useful, but its ability to fully capture and analyse dynamic behaviours inherent in modern safety-critical systems is largely unsolved. There are various techniques developed to enhance the capabilities of classical FTA. Some of these techniques are used for qualitative analysis, others for quantitative analysis and some others for both qualitative and quantitative analysis. Pandora is a more recent technique formulated to incorporate sequential failure logic (SFL) into FTA. It does so by introducing new temporal gates and laws for logically analysing safety-critical systems.

The difference between Pandora and other techniques is that it provides novel logical gates and laws which eliminate some contradictions present in the others. This will be discussed further in later chapters. However, Pandora is only a logical analysis technique. The question remains: how can Pandora be used as a quantitative tool to evaluate state-of-the-art safety-critical systems which feature dynamic behaviour due to their multiple modes of operation? This is the motivation of this thesis.

The solution to this question requires the formulation of rigorous techniques for probabilistically analysing Pandora's new gates, which are required for the evaluation of system failure probabilities. The evaluation of system failure provides a lot of benefits to engineers and system designers who use it. They are able to prioritise components in relation to their contribution to the occurrence of the system failure which aids them in making strategic decisions concerning the reliability of the system.

1.3 Research Question

The question this thesis aims to answer is:

How can the temporal fault tree of a safety-critical system featuring various failure distributions be quantitatively analysed using both simulation and analytical approaches?

The following objectives are formed to address the research question.

Probabilistic Evaluation of Each Novel Temporal Gate

To perform the probabilistic analysis of any fault tree, whether a traditional fault tree or a dynamic fault tree, one needs to identify how to evaluate probabilistically each gate in the fault tree. Each gate has a specific description and behaviour both logically and semantically. This necessitates the probabilistic evaluation of each individual gate.

The initial objective of this research is to identify or formulate techniques for probabilistically evaluating the Priority-AND (PAND), Simultaneous-AND (SAND) and Priority-OR (POR) gates as defined by Pandora. The evaluation of PAND, SAND and POR will contribute significantly to the determination of the overall system failure probability.

Probabilistic Evaluation of MCSQs in Temporal Fault Trees

As important as evaluating individual temporal fault tree gates (POR, PAND, SAND) probabilistically is, it is not sufficient when evaluating MCSQs with two or more different gates. It is an objective of this research to identify or formulate techniques for evaluating MCSQs and techniques for parsing and evaluating such expressions computationally. This thesis also investigates how MCSQs with components of various distributions can be evaluated using simulation approaches.

Techniques to Evaluate the Top-event Probability

The top-event probability of a system, also referred to as the system failure probability, is the probability that a system failure will occur within a given time duration. Formulating techniques of calculating this probability is the ultimate objective of this research. This research investigates the possibility of evaluating the top-event probabilities either analytically or with simulation or a combination of both. The possibility of calculating importance measures – measures of how contributing combinations of events influence a system failure – is also considered.

1.4 Research Contribution

Succinctly, this thesis provides both analytical and simulation techniques for quantitatively evaluating state-of-the-art safety-critical systems. The proposed solutions are scalable; they enable the determination of component importances and potential applications to other fields of study other than engineering. In details, the thesis contributions are as follows:

Individual Gate Evaluation

PAND gates can be considered to be inclusive (iPAND) or exclusive (ePAND). In the former, input events occur one after another or at the same time to trigger the occurrence of the output gate whilst in the later the events need to occur strictly one after another to trigger the output event. Various techniques to evaluate the iPAND gate have been identified. These include a derivation from Calculus, Markov Chains, Stochastic PetriNets, Bayesian Networks, Monte Carlo simulation and Poisson Stochastic Process.

It has been proven that the iPAND and ePAND gates are probabilistically evaluate to zero, however, they have different logically interpretations. The expression from Calculus has been adopted for use in Pandora.

An expression for the SAND gate has been formulated from three of Pandora Laws and from logical analysis. However, this expression evaluates to zero; meaning for any statistically independent exponentially distributed events, the probability of the events occurring is zero. An expression for the POR gate has been formulated from Markov Chains, Calculus and Pandora's logical analysis. A simulation solution has also been developed.

The SAND gate is used to model the situation where two or more events occur at exactly the same time. It is known (Merle and Roussel, 2010) that the occurrence of two exponentially distributed and statistically independent events is zero. However, there exist nearly-simultaneous situations where two or more input events trigger the occurrence of an output event if the input events occur within a relatively small time interval. This necessitates the description and the formulation of a mathematical expression for a logical gate to represent such scenarios. The proposed gate is called parameterised SAND or pSAND. pSAND is similar to SAND but has a consideration for a time interval within which all input events must occur for the output event to occur. Analytical and simulation solutions have been formulated for probabilistically evaluating the pSAND gate.

Probabilistic Evaluation of MCSQs in Pandora

With the help of a precedence order, the evaluation of MCSQs has been made possible. To implement MCSQs evaluation computationally, the Shunting-Yard algorithm is slightly modified to parse MCSQs into Reverse Polish Notation (RPN). A postfix algorithm is formulated to evaluate the probability of the expression generated from the RPN. The evaluation of MCSQ probability is very important for the evaluation of the system failure probability. Conditions necessary for Monte Carlo simulations of MCSQs with various component failure distributions have been constructed; these enable the modelling and simulation of total system failures.

Probabilistic Evaluation of System Failure

Due to the probabilistic evaluation of Pandora's temporal gates in combination with traditional Boolean gates, the system failure probability of temporal fault trees has been made possible. It also makes the evaluation of importance measures possible. These do

not only give information about the probability of a system failure within a time frame but also makes possible the ranking of system components with respect to their contribution to the occurrence of the system failure. This is the focal contribution of this research and can be succinctly stated as: the probabilistic analysis of dynamic safety-critical systems.

Probabilistic Evaluation of System with Other Distributions

All analytical solutions described in this thesis are restricted to exponential distribution. However, some state-of-the-art systems feature components with other failure distributions such as the Weibull or Lognormal distributions. In this thesis, Monte Carlo models have been constructed for evaluating systems with exponential, Weibull and Lognormal distributions. These have been made possible by the use of the time-to-failure (TTF) of components to model simulation conditions where temporal gates are present.

Alternative Qualitative Analysis

In addition to contributing to the quantitative analysis of temporal fault trees, this thesis also provides two techniques to enhance the qualitative analysis of temporal fault trees. The first, known as Groups and Modules Modularization Technique (GMMT), is an enhancement of existing qualitative techniques by the use of modularisation and the second, known as Temporal Binary Decision Diagram (TBDD), is a novel technique for extending Binary Decision Diagrams to include temporal features.

1.5 Assumptions

The following assumptions are made in this thesis.

Statistical Independence

It is assumed that the occurrences of basic events are statistically independent, meaning that the occurrence of an event is not dependent on the occurrence of another event in any way.

Non-Repairable Events

It is assumed that events are non-repairable. That is, once a basic event has occurred, it remains in that state. Therefore the state of an event can only change from ‘not occurred’ (false) to ‘occurred’ (true) and not vice versa.

System Coherency

The final assumption in this thesis is that all systems are coherent. A system is coherent “if sufficient components are functioning to cause the system to function, the functioning of additional components can only improve matters; if sufficient components have failed to cause system failure, then the failure of additional components can only make matters worse” (Esary and Proschan, 1963).

1.6 Structure

The structure of this thesis is arranged to give the reader a progressive understanding of the background, motivation, contribution and conclusions. The structure, organised into seven chapters, is as follows:

Chapter 1: Introduction

This introductory chapter explains the central problem – the probabilistic evaluation of fault trees of dynamic safety-critical systems – and sets out how the problem will be solved. It also discusses the specific aims and objectives of the work.

Chapter 2: Background

This chapter provides the theoretical background underpinning the remaining chapters of the thesis. It commences with the definition, importance, and terminology of reliability engineering, and introduces fundamental probability theories. It then discusses Fault Tree Analysis (FTA), its advantages and disadvantages and why it needs to be enhanced. It goes on to discuss two enhancements of FTA – Dynamic Fault Trees (DFTs) and Pandora. Finally, a detailed review of significant quantitative techniques is made.

Chapter 3: Temporal Quantitative Analysis

In this chapter, the theoretical bases of the quantitative analysis of dynamic systems are made. These include the algebraic, timing and probabilistic models for all OR, AND, POR, PAND, SAND and parameterised-SAND (pSAND) gates used in Pandora. It also provides analytical algorithms for evaluating MCSQs leading to the evaluation of system failure probability.

The analytical approach is restricted to the exponential distribution. In concluding this chapter, models of how Monte Carlo simulations for constructing MCSQs are discussed. The Monte Carlo simulations are not restricted to only exponential distributions. Discussions on Monte Carlo simulations for Weibull and Lognormal distributions are made.

Chapter 4: Temporal Qualitative Analysis

Though the major aim of this thesis is the provision of quantitative techniques for the evaluation of temporal fault trees, a technique based on modularisation for enhancing Pandora's qualitative technique has been developed. This chapter contains a modularization technique and an alternative technique for qualitatively analysing temporal fault trees using Binary Decision Diagrams (BDDs).

Chapter 5: Case Study

This chapter demonstrates the use of developed techniques in the analysis of an Aircraft Fuelling System (AFS). AFS describes how a dual engine aircraft can be fuelled from five tanks by the use of connecting valves, flow meters, sensors and the likes. It also demonstrates how quantitative and qualitative analyses techniques developed in chapters 3 and 4 can be applied to a temporal fault tree derived from AFS's description. The results are top event occurrence probabilities for various system lifetimes using various component failure distributions and importance measures.

Chapter 6: Evaluation

Firstly, this chapter discusses the achievements of this thesis against its aim and objectives. Secondly, it provides a comparative analysis of proposed techniques will be made against each other to evaluate them. Discussions on the successes and limitations of this research are also made and finally, directions for future research are identified.

Chapter 7: Conclusion

This is the final chapter that summarises the work discussed in this thesis: problems this thesis aims to solve, how it has solved them and why this thesis is necessary.

1.7 Note on Publication

It should be noted that some materials of this thesis have been published in a number of academic outputs. Materials in chapter three has been published in Edifor *et al.* (2012), Edifor *et al.* (2013) and two other articles has been submitted for publications. Chapter four has been published in Edifor *et al.* (2013); another article has been extracted from this chapter and submitted for publication.

Chapter Two

BACKGROUND

This chapter discusses theories underpinning the content of this thesis by providing definitions and some fundamental probability theories of reliability engineering. It also discusses various techniques for logically and probabilistically evaluating fault trees. The differences between these techniques are clearly distinguished and their strengths and weaknesses are reviewed. Finally, justifications for using some of these techniques are made.

2.1 Reliability Engineering

A **system** is an entity made up of a set of logically connected components performing specific tasks to enable the entity to perform and provide its overall functionality (Stapelberg, 2009). In this context, this means a system is one which takes in a set of inputs and converts them into required outputs by the use of its complicated web of logically related components (Naikan, 2009). A **component** or subsystem is an integral constituent of a system that contributes to the overall functionality of the system (Stapelberg, 2009; Naikan, 2009). In more detail, a subsystem can become a system when considered as an entity under study. For example, a car can be a system with an engine as a sub-system. However, at a lower level of abstraction, an engine can be a system with various components, such as an alternator as a subsystem.

A system is said to be reliable if it can be trusted (OUP, 2013). The US Department of Defense defines **reliability** as “the probability that an item can perform its intended function for a specified interval under stated conditions” and **reliability engineering** as “the technical discipline of estimating, controlling, and managing the probability of failure in devices, equipment and systems” (DoD, 1998). Meaning, the reliability R of a system is the probability that the system will perform the functions it was designed to perform in the conditions it has been designed to function within a given time interval $\{0, t\}$; where 0 is the start time and t is the system’s lifetime. Conversely, system **unreliability**, Q , is the probability that a system will fail to perform its intended functions under normal conditions within a given time interval $\{0, t\}$.

Reliability and **safety** are sometimes used interchangeably but they are technically not the same. **Safety** is the ability of a system not to cause unacceptable harm or risks to human life and its environment (BSI, 1979). It is commonly assumed that the increase of a system's (or its components') safety is directly proportional to its reliability. Leveson (2011) debunks this assumption: a system can be safe yet not necessarily reliable and vice versa. For example, a nuclear reactor may be reliable but that does not make it safe; it may pollute its surrounding environments. Conversely, a personal computer may be very unreliable but it may be very safe to use. Generally, though, unreliability makes safety-critical systems less safe.

The reliability of a system can be seriously compromised by a failure, which is usually caused by a fault. A system **failure** occurs when the system deviates from its specified performance – it is unable to perform its intended function as specified by the system designer (Pukite & Pukite, 1998; Sundararajan, 1991). A **fault** can be described as an erroneous state of a component (Pukite & Pukite, 1998) or “an abnormal undesirable state of a system ... induced by [the] presence of an improper command or absence of a proper one, or by a [component] failure” or “a non-compliance with specifications” (Sundararajan, 1991). A very simple example to distinguish the difference between a failure and a fault is this:

Failure: A battery-powered torch will have a failure if the torch fails to produce light under the specified conditions it was manufactured to operate.

Fault: A failure of a torch could be caused by wrongly placing its batteries.

A fault can be classified into one of three basic categories: primary, secondary or command (Bedford & Cooke, 2001). Primary faults occur when a component fails to deliver the intended tasks for which it was designed under suitable conditions for which it was designed: an example is wrongly placing the batteries of a torch. A secondary fault however, arises when the component fails to deliver the intended tasks for which it was designed under unsuitable conditions for which it was not designed: for example using a non-water resistant torch in a swimming pool. A command fault, however, occurs when the component functions in the correct manner in which it was designed but produces wrong output or functions at a wrong time or place. For example, there will be a command failure if a torch manufactured to produce light only when its switch is turned on produces light when its switch is turned off (when the torch's switch is short circuited).

While secondary and command faults are caused by external entities, primary faults are caused by the component itself. An **error** is a basically the “manifestation of a fault” (Pukite & Pukite, 1998).

2.1.1 Reliability Theory

A system that is prone to fail frequently within a given duration is deemed less reliable than another that fails less often within the same duration if both systems have the same intended system lifespan. Reliability can therefore be expressed as a function of time and can be probabilistically evaluated. Just as any probability function, the reliability R of a system S over a period of time t is a numeric value that lies between 0 and 1 inclusive. The more the probability approaches zero, the lower the reliability; the more it approaches 1 the higher the reliability. Therefore a reliability of 1 means the system is 100% reliable – the system cannot fail at any time – which is unrealistic. If the reliability is zero then the system is always in a failed state – it never functions as intended. The mathematical relationship between reliability, R , and unreliability, Q is popularly known and given as:

$$R(S)\{t\} = 1 - Q(S)\{t\} \quad (2.1-1)$$

Where $0 < t < \infty, 0 \leq R(S)\{t\} \leq 1$ and $0 \leq Q(S)\{t\} \leq 1$

The **failure rate** λ of a system S between the intervals t_1 and t_2 is defined as “the ratio of probability that failure occurs in the interval, given that it has not occurred prior to t_1 , the start of the interval, divided by the interval length” (DoD, 1998) and is mathematically expressed as:

$$\lambda(S)\{t\} = \frac{R(S)\{t_1\} - R(S)\{t_2\}}{(t_2 - t_1)R(S)\{t_1\}} \quad (2.1-2)$$

In simple terms, a failure rate is the rate at which a component fails given a time interval. The failure rate of a system is generally depicted with a bathtub like curve (Henley & Kumamoto, 1981; DoD, 1998) as seen in Figure 2.1-1. It is assumed that due to poor design, lack of appropriate structures, procedures in the manufacturing process or use of low quality components, the failure rate of a system is initially high. This is known as the infant mortality stage. However, as the errors, mistakes, omissions etc., are corrected, the system gets into a more stable operation zone where the failure rate remains at its presumably lowest ebb for a period of time. This period of the system’s lifetime is called the useful life. As the system ages, it enters its wear out period and the failure rate begins to increase again.

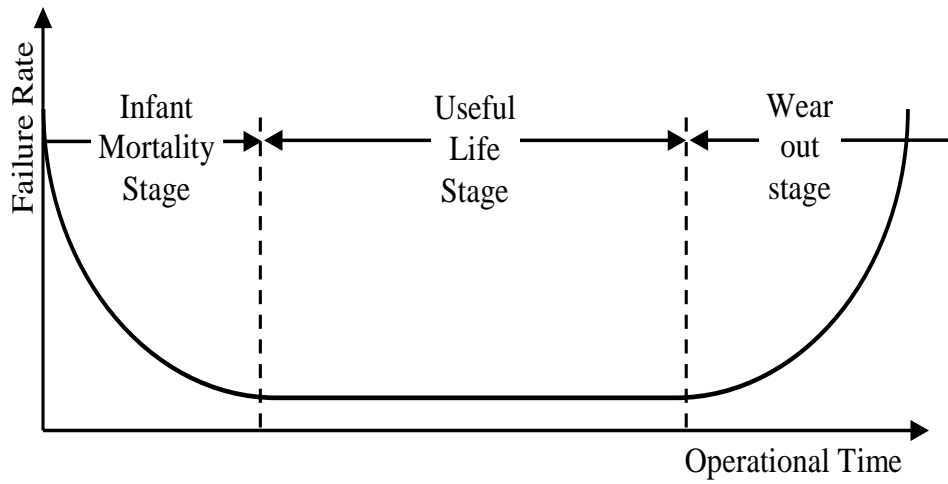


Figure 2.1-1: Failure rate curve

The hazard rate h of a system S is also known as the instantaneous failure rate. It is the value of the failure rate as the system time interval length approaches zero (DoD, 1998) and is given as:

$$h(S)\{t\} = \lim_{\Delta t \rightarrow 0} \left[\frac{R(S)\{t_1\} - R(S)\{t_2\}}{(t_2 - t_1)R(S)\{t_1\}} \right] = \frac{1}{R(S)\{t\}} \left[-\frac{dR(S)\{t\}}{dt} \right] \quad (2.1-3)$$

From 2.1-3 is evident that

$$\frac{1}{R(S)\{t\}} \left[\frac{dR(S)\{t\}}{dt} \right] = -h(S)\{t\}; \therefore \frac{dR(S)\{t\}}{R(S)\{t\}} = -h(S)\{t\}d(t) \quad (2.1-4)$$

$$\int_0^t \frac{dR(S)\{t\}}{R(S)\{t\}} = - \int_0^t h(S)\{t\}d(t)$$

$$\ln R(S)\{t\} - \ln R(S)\{0\} = - \int_0^t h(S)\{t\}d(t)$$

$$\ln R(S)\{t\} = - \int_0^t h(S)\{t\}d(t)$$

However, at time $t = 0$, $R(S)\{0\} = 1$ and

$$R(S)\{t\} = \exp \left[- \int_0^t h(S)\{t\}d(t) \right] \quad (2.1-5)$$

The next time a system is expected to fail is referred to as the time-to-failure (TTF). Technically, the average TTF is called the Mean TTF (MTTF) and is given by:

$$MTTF = \int_0^{\infty} R(S)\{t\}dt \quad (2.1-6)$$

The importance of reliability engineering cannot be over emphasised. In 1986, the Challenger space shuttle blew up a few minutes after take-off because of some leak through a faulty seal, killing seven people (BBC, 2009b). Three months later, a nuclear disaster (Chernobyl) caused by a power surge (due to design flaw) led to massive explosions which blew the top off the reactor in the Ukraine. It is estimated that Chernobyl could potentially cause the death of over nine thousand people (BBC, 2009a). A more recent explosion leading to a massive disaster was the BP oil spill at the Gulf of Mexico. Even though people perished on the spot, the extent of the damage cannot be accurately quantified since it directly affects aquatic life, which indirectly affects human life. According to BP (2010) the disaster was caused by “a sequence of failures involving a number of different parties ...” arising from “a complex and interlinked series of mechanical failures, human judgments, engineering design, operational implementation...”

These disasters represent a few examples; many other system disasters have occurred causing a lot of harm to man, plants animals and entire ecosystems. For these reasons, there is a need for rigorous reliability analysis of safety-critical systems.

2.1.2 Probability Theory

Every system is doomed to eventually fail sometime in the future because no system is perfectly reliable forever. For this reason, engineers must anticipate the failure of the systems they design and make appropriate adjustments to reduce the frequency of these failures. The more a system is liable to fail, the less trusted it becomes and the less likely it will be used.

Uncertainties are part of normal life. Statements such as, “I am not too sure”, “it is very likely to rain today”, etc. are made almost every day. Even though unaware to many people, they make several probabilistic statements every day. In reliability engineering, probabilistic techniques are used in evaluating a system’s reliability. Unlike the vague, ambiguous probabilistic statements made by people every day, reliability engineering requires accurate and precise probabilistic evaluations. This must be so for various reasons including:

1. the lives of persons who will be using the systems and safety of the system's environment (including people, properties etc.) are at stake
2. frequency of failures must be known because repairs cost money
3. entire system reliability must be known to boost business competitive advantage.

According to Freund (1962), mathematical probabilistic evaluation began in the 17th century when a gambler, Chevalier de Mere, consulted the famous mathematician, Blaise Pascal, about how to divide the winnings in a game of chance. Blaise's correspondence with Pierre Fermat, a lawyer and mathematician, is what produced the origin of modern probability theory. Later on, other mathematicians further developed the theory and applied it to various fields apart from gambling (Shooman, 2002).

Generally, probability can be evaluated in three main ways: the Priori or equally-likely-events approach, the relative-frequency approach and using axiomatic definition (Papoulis & Pillai, 2002). The priori approach is convenient for evaluating the probability of equally-likely-events but cannot handle not-equally-likely events. The relative-frequency approach can be used in evaluating unequally likely events; however, it can lead to theoretical problems if care is not taken. The axiomatic approach is recognised as superior to the priori and relative-frequency approaches because it eliminates ambiguities whilst providing robust foundation for complex applications (Papoulis & Pillai, 2002).

This thesis focuses on the axiomatic approach. An event A with probability of occurrence, P satisfies the following conditions:

1. the probability is a non-negative number but no greater than 1

$$0 \leq P(A) \leq 1 \quad (2.1-7)$$

2. if Ω is the whole or entire set of events, the probability of the whole set is unity

$$P(\Omega) = 1 \quad (2.1-8)$$

3. if \emptyset is the empty set, A and B are independent events, $A \cap B = \emptyset$, then

$$P(A \cap B) = P(A).P(B) \quad (2.1-9)$$

An event A is said to be statistically dependent on another event B if the occurrence of B affects the probability of the occurrence of A .

The dependent or conditional probability of A given that B has occurred, $P(A/B)$, is generally given by:

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \quad (2.1-10)$$

$$P(A \cap B) = P(A|B)P(B)$$

Conversely, A and B are said to be statistically independent if the occurrence of B does not affect the probability of the occurrence of A . If the probability of A and B occurring are $P(A)$ and $P(B)$ respectively, then from 2.1-10 the probability of A occurring given that B has occurred will be the same as the probability of A .

A and B are said to be mutually exclusive or disjoint if they cannot occur at the same time. That is:

$$P(A \cap B) = 0$$

On the other hand, A and B are not mutually exclusive if they can both occur at the same time:

$$P(A \cap B) \neq 0$$

Probability Distribution

A variable X can be defined as a **random variable** if each value of X can be associated with an element x in an event A defined on a sample space S (Shooman, 2002). Meaning, a random variable is a variable whose value is not fixed or single but can assume a set of different possible values – each value having an occurrence probability associated with it. If X assumes a finite number of values, then it is termed as a **discrete random variable**. Random variables can be defined over a continuum (real number) that is not discrete; this is a **continuous random variable**. Unlike the discrete random variables where the value of a random variable is finite, the values for a continuous random variable cannot be counted.

The **probability distribution function** (pdf) (Dinov, 2004) of a continuous random variable X is a function $f(x)$, such that for any two numbers a and b ,

$$P(a \leq X \leq b) = \int_a^b f(x)dx \quad (2.1-11)$$

The **cumulative distribution function** (cdf) (Dinov, 2004) of a continuous random variable X is a function $F(x)$, such that for every x ,

$$F(x) = P(X \leq x) = \int_{-\infty}^x f(y)dy \quad (2.1-12)$$

Therefore for every continuous random variable X , for every element x , if there exist the derivative $F'(x)$ then

$$F'(x) = f(x)$$

Probability distributions have different properties and exhibit different behaviours. For this reason, each has its distinct functions (PDF, CDF etc.). Table 2.1-1 to Table 2.1-3 are summaries of the common life distributions (O'Connor, 2011) used in reliability engineering.

Table 2.1-1: Summary of Exponential Distribution (O'Connor, 2011)

Function	Description
Parameters	λ = hazard rate; S = system; t = system lifetime
$f(S)\{t\}$	$f(S)\{t\} = \lambda e^{-\lambda t}$ (2.1-13)
$F(S)\{t\}$	$F(S)\{t\} = \int_0^t f(S)\{x\}dx = \int_0^t \lambda e^{-\lambda x} dx = 1 - e^{-\lambda t}$ (2.1-14)
$h(S)\{t\}$	$h(S)\{t\} = \lambda$ (2.1-15)
$R(S)\{t\}$	$R(S)\{t\} = 1 - F(S)\{t\} = 1 - (1 - e^{-\lambda t}) = e^{-\lambda t}$ (2.1-16)
Application	Used to model no-wear out or cumulative damage situations, some electronic components such as capacitors or integrated circuits (ICs) and random shocks.

Table 2.1-2: Summary of Lognormal Distribution (O'Connor, 2011)

Function	Description
Parameters	S = system; t = system lifetime μ_N = mean of normally distributed $\ln(x)$. $\mu_N = \ln\left(\frac{\mu^2}{\sqrt{\sigma^2 + \mu^2}}\right)$ σ_N = standard deviation of normally distributed $\ln(x)$. $\sigma_N = \ln\left(\frac{\sigma^2 + \mu^2}{\mu^2}\right)$

	ϕ = standard normal PDF; Φ = standard normal CDF
$f(S)\{t\}$	$f(S)\{t\} = \frac{1}{\sigma_N t \sqrt{2\pi}} e^{\left[-\frac{1}{2}\left(\frac{\ln(t)-\mu_N}{\sigma_N}\right)^2\right]} = \frac{1}{\sigma_N t} \phi\left[\frac{\ln(t)-\mu_N}{\sigma_N}\right]$ (2.1-17)
$F(S)\{t\}$	$F(S)\{t\} = \frac{1}{\sigma_N t \sqrt{2\pi}} \int_0^t \frac{1}{u} e^{\left[-\frac{1}{2}\left(\frac{\ln(u)-\mu_N}{\sigma_N}\right)^2\right]} du = \Phi\left(\frac{\ln(t)-\mu_N}{\sigma_N}\right)$ (2.1-18)
$h(S)\{t\}$	$h(S)\{t\} = \frac{\phi\left(\frac{\ln(t)-\mu_N}{\sigma_N}\right)}{\sigma_N t \left(1 - \Phi\left[\frac{\ln(t)-\mu_N}{\sigma_N}\right]\right)}$ (2.1-19)
$R(S)\{t\}$	$R(S)\{t\} = 1 - F(S)\{t\} = 1 - \Phi\left(\frac{\ln(t)-\mu_N}{\sigma_N}\right)$ (2.1-20)
Application	Suitable for modelling many life distributions and components with repair time distributions. It is also appropriate for parameter variability and modelling elements in breakage processes.

Table 2.1-3: Summary of Weibull Distribution (O'Connor, 2011)

Function	Description
Parameters	S = system; t = system lifetime α = scale parameter; β = shape of the distribution or slope
$f(S)\{t\}$	$f(S)\{t\} = \frac{\beta t^{\beta-1}}{\alpha^\beta} e^{-\left(\frac{t}{\alpha}\right)^\beta}$ (2.1-21)
$F(S)\{t\}$	$F(S)\{t\} = 1 - e^{-\left(\frac{t}{\alpha}\right)^\beta}$ (2.1-22)
$h(S)\{t\}$	$h(S)\{t\} = \frac{\beta}{\alpha} \left(\frac{t}{\alpha}\right)^{\beta-1}$ (2.1-23)
$R(S)\{t\}$	$R(S)\{t\} = e^{-\left(\frac{t}{\alpha}\right)^\beta}$ (2.1-24)
Application	It is the most popular life distribution used in reliability engineering. Used for modelling acceptance sampling, warranty analysis, wear modelling, corrosion modelling maintenance and renewal etc.

Component Input Data

As discussed earlier, a total system failure is a result of combinations of basic event (component) failures. Therefore, before the probability of a system failure can be determined, the probability of the failure of the various combining components leading to the system failure must be known. There are various failure-parameters supplied for components for the evaluation of the total system failure occurrence. These include component failure probability, event occurrence probability and pure event probability.

The component failure probability, P , is also known as the component unreliability. Where t and λ are the relevant time and failure rate of a component C respectively, P , t and λ are related in (2.1-25) for exponentially distributed events (Vesely *et al.*, 2002).

$$P(C)\{t\} = 1 - e^{-\lambda t} \quad (2.1-25)$$

Where d is the functional duty cycle: total operation time divided by total mission time. If λ_o is the component failure rate in the operating state and λ_N is the contribution to the component failure rate from the non-operating state then the failure rate of the component, λ_c , is:

$$\lambda_c = \lambda_o d + \lambda_N(1 - d) \quad (2.1-26)$$

The event occurrence probability is very similar to the component failure probability. However, with the event occurrence probability, the inputs supplied are the event occurrence rate, instead of component failure rate, and time. Supposing the event occurrence rate and time of an event E are respectively given as λ_e and t , then the event occurrence probability P_e can be given as (Vesely *et al.*, 2002):

$$P(E)\{t\} = 1 - e^{-\lambda_e t} \quad (2.1-27)$$

The pure event probability, according to Vesely *et al.* (2002) is also known as the probability per act or probability per demand. It is generally an input for an event “for which a failure rate or occurrence rate per unit time is not recorded” (Vesely *et al.*, 2002). A typical example is an input for human error.

2.2 Fault Tree Analysis (FTA)








FTA (Vesely *et al.*, 1981) is a safety analysis technique used for evaluating the reliability of safety-critical systems since the early 1960s. It was first conceived when Bell Laboratories, in collaboration with the US Air Force, examined the reliability of one of the Air Force's weaponry systems. The Boeing Company identified the usefulness of this tool and adopted it for use. Since then FTA has gained worldwide popularity and has been used in various fields beyond engineering, such as supply chain (Ziegenbein & Baumgart 2006), health care (Ong & Coiera 2010), service process (Geum *et al.* 2009) and agriculture (Buck 1997) etc.

As a safety analysis technique, FTA falls under a broader group of techniques known as risk assessment techniques. Risk assessment techniques are methods used to identify, manage, control and evaluate situations of a system that have the potential of putting the system in an unwanted state (Garvey, 2008). Detailed discussion on risk assessment techniques can be found in Alverbro *et al.* (2010) and Izvercian *et al.* (2012).

A fault tree in reliability engineering is a graphical structure consisting of events related by logical connectives to describe how combinations of these events can lead to the failure of a system. To construct the fault tree of a system, a system failure is first decided. The system failure is examined to determine its immediate causes; each combination of causing events is related with appropriate logical gates. All immediate causes are also examined to determine combinations of events that can cause them. This identification of immediate causes events continue until all causing events are basic events. A detailed review of these can be obtained from Vesely *et al.* (2002).

FTA revolves around the concept of events. An **event** is a binary outcome of a component's functionality: whether it has failed or it is operational. If an event's outcome contributes to the occurrence of another event then the causing event is known as an input event whilst the resulting event is known as an output event. A basic event is an event that has no input events but can be an input event itself. An intermediate event is a combination of events that has one or more input events and is by itself an input event to another event. A top event corresponds to a system failure; it usually has input events but cannot have an output event. Events are combined using logic gates. A logic gate is a logic operator that transforms its input events into an output event based on its (gates) behaviour. A logic gate could be unary, binary or multi-nary. FTA uses a number of logic gates; a summary of these gates (Vesely *et al.*, 2002) are described in Table 2.2-1.

Table 2.2-1: FTA's Logical Gates

Name of Gate	Abbr.	Symbol (analysis)	Symbol (graphical)	Description
AND	AND	.		Output event occurs if all of its inputs events occur
OR	OR	+		Output event occurs if at least one of its inputs events occur
COMBINATION	NA	NA		Output event occurs if a number, n , of its inputs events occur
EXCLUSIVE OR	NA	NA		Output event occurs if exactly one of its inputs events occur
PRIORITY AND	NA	NA		Output event occurs if all of its input events occur in a specific sequence: one after another
INHIBIT	NA	NA		Output event occurs if its input event occurs (based on the satisfaction of a condition)
K-OUT-OF-N	NA	NA		Output event occurs if at least K input events occur out of N input events.

We describe the Boolean AND and OR gates (below) because they are the most relevant to this thesis. They are popular gates employed for logical and probabilistic analyses of classical fault trees.

AND Gate

A logical AND gate is a conjunctive operator that is fired when all its input events have occurred. All input events must occur to trigger the AND gate. There is no specification of the relative times within which the events occur; sequential failures of events are ignored. Figure 2.2-1A is a graphical model of an AND gate with two events X_1 and X_2 ; Q is fired when X_1 and X_2 have both occurred. Figure 2.2-1B is an AND with n events; Q is triggered when all input events, $X_1, X_2, \dots, X_{n-1}, X_n$ have occurred.

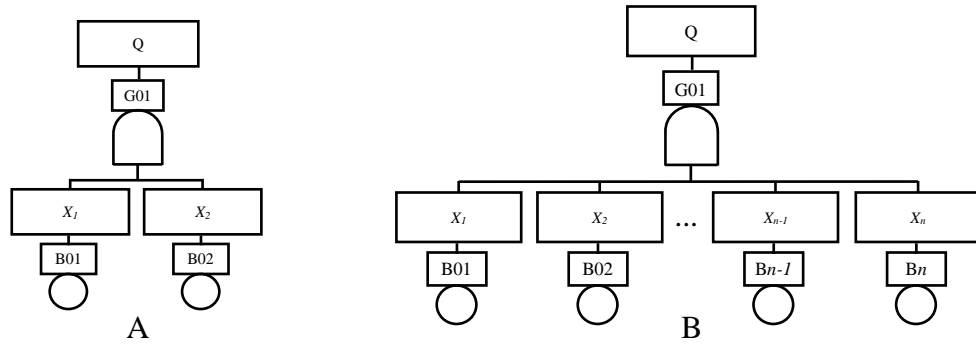


Figure 2.2-1: Graphical representations of the AND gate

Figure 2.2-2 is a timing diagram for Figure 2.2-1A. In Figure 2.2-2A X_1 occurs before X_2 , $X_1 \cdot X_2$ occurs at $t(X_2)$. In Figure 2.2-2B X_1 occurs at the same time as X_2 , $X_1 \cdot X_2$ occurs at $t(X_1)$ or $t(X_2)$. In Figure 2.2-2C X_2 occurs before X_1 , $X_1 \cdot X_2$ occurs at $t(X_1)$. The AND gate occurs when all its input events have occurred.

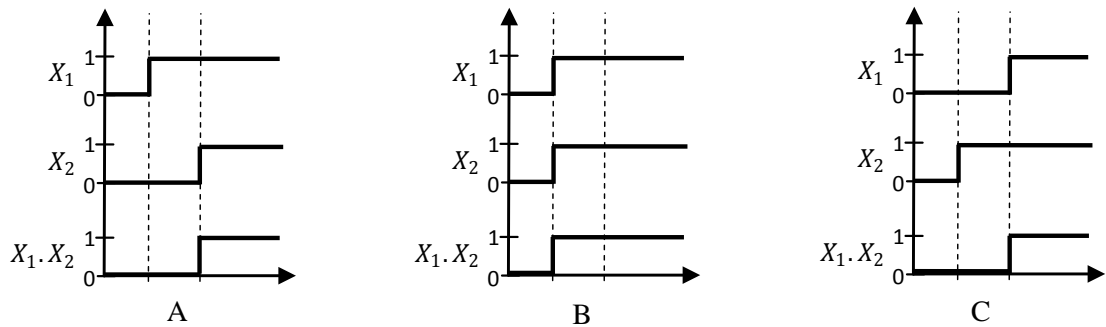


Figure 2.2-2: Timing diagram of the AND gate with 2 input event

If t is the lifetime of the event in Figure 2.2-1A, (2.2-1), (2.2-2) and (2.2-3) are expressions corresponding to Figure 2.2-2 A, B and C. This means, in (2.2-1) the time of occurrence of X_1 is less than the time of occurrence of X_2 ; this means X_1 occurs before X_2 . It must be noted that the symbol ' $<$ ' used in (2.2-1) means less than (not Priority-AND as discussed later). In (2.2-2) both events occur at the same time. X_2 occurs before X_1 in (2.2-3).

$$t(X_1) < t(X_2) \Rightarrow t(X_1 \cdot X_2) = t(X_2) \quad (2.2-1)$$

$$t(X_1) = t(X_2) \Rightarrow t(X_1 \cdot X_2) = t(X_2) \quad (2.2-2)$$

$$t(X_1) > t(X_2) \Rightarrow t(X_1 \cdot X_2) = t(X_1) \quad (2.2-3)$$

OR Gate

The logical OR gate is a disjunctive operator that models the occurrence of events; all input events do not need to occur for the OR gate to be fired. The OR gate is fired only

if at least one of the input events have occurred. Just as the AND gate, there is no specification of the relative times within which the input events of an OR gate occur; particular sequences do not matter – at least one must occur. Figure 2.2-3A is a graphical representation of an OR gate with two events; Q is fired when at least X_1 or X_2 have both occurred. Figure 2.2-3B is OR with n events; Q is triggered when at least one of its input events have occurred.

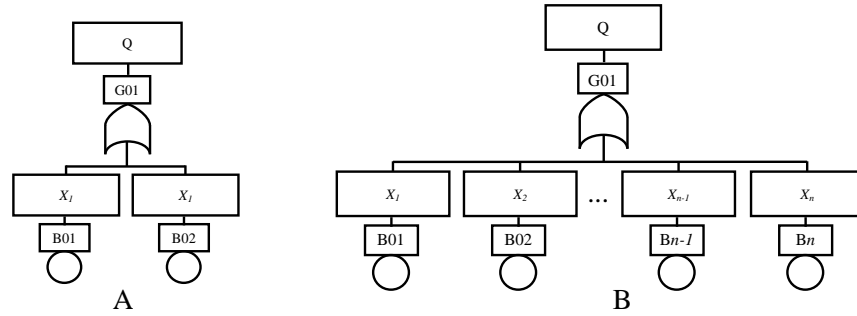


Figure 2.2-3: Graphical representations of the OR gate

Figure 2.2-4 is a timing diagram for Figure 2.2-3A. In Figure 2.2-4-A X_1 occurs before X_2 , $X_1 + X_2$ occurs at $t(X_2)$. In Figure 2.2-4B X_1 occurs at the same time as X_2 , $X_1 + X_2$ occurs at $t(X_1)$ or $t(X_2)$. In Figure 2.2-4C X_2 occurs before X_1 , $X_1 + X_2$ occurs at $t(X_1)$. The OR gate occurs when at least one of its input event occurs.

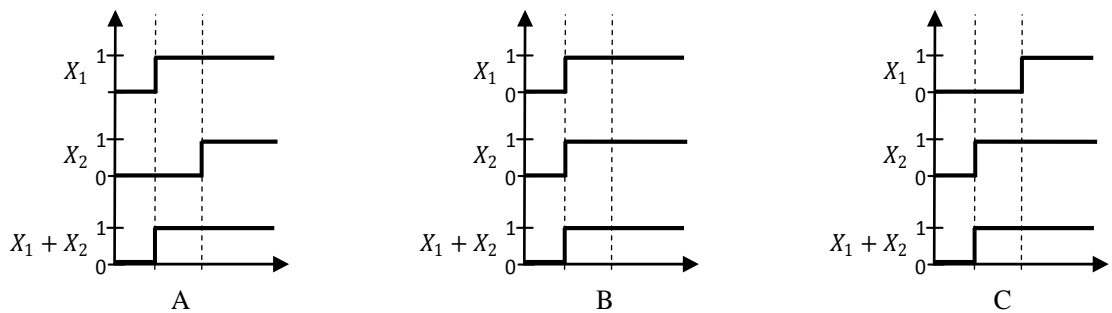


Figure 2.2-4: Timing diagram of the OR gate with 2 input event

If t is the lifetime of the system in Figure 2.2-3A, (2.2-4), (2.2-5) and (2.2-6) are expressions corresponding to Figure 2.2-4 A, B and C respectively.

$$t(X_1) < t(X_2) \Rightarrow t(X_1 + X_2) = t(X_1) \quad (2.2-4)$$

$$t(X_1) = t(X_2) \Rightarrow t(X_1 + X_2) = t(X_1) \quad (2.2-5)$$

$$t(X_1) > t(X_2) \Rightarrow t(X_1 + X_2) = t(X_2) \quad (2.2-6)$$

Again, the symbol ' $<$ ' used in (2.2-4) means less than. In (2.2-4) the time of occurrence of X_1 is less than the time of occurrence of X_2 ; this means X_1 occurs before X_2 . In (2.2-5) both events occur at the same time. X_2 occurs before X_1 in (2.2-6).

To clarify the construction of fault trees, consider the HPS scenario described in Chapter 1. The top-event is a total failure in a medical device (MD). A failure in MD is caused by an internal failure of MD or an omission of power supply from the electricity system (E) and the generator system (G). Though an internal failure in MD is a basic event – due to our level of abstraction, it cannot be broken down further – omission of power supply from E or G is an intermediate event with two non-basic events so each needs investigating further. An omission of power from the electricity system is caused by a failure in the electricity system or lack on an input command. An internal failure of the generator or an internal failure of the sensor or an absence of input demand causes an omission from the generator. If X_f stands for the output failure of an event X , X_i for internal failure of X and X_o for omission of output from X then the above description of HPS can be expressed as:

$$MD_f = MD_i \text{ OR } E_o \text{ AND } G_o$$

$$E_o = E_i \text{ OR } I_o$$

$$G_o = G_i \text{ OR } S_i \text{ OR } I_o$$

From these logical expressions, a fault tree, Fig. 2.2-5, can be generated using the FTA symbols in Table 2.2-1 – where GXX is the number of the gate and BXX is the number of the basic event.

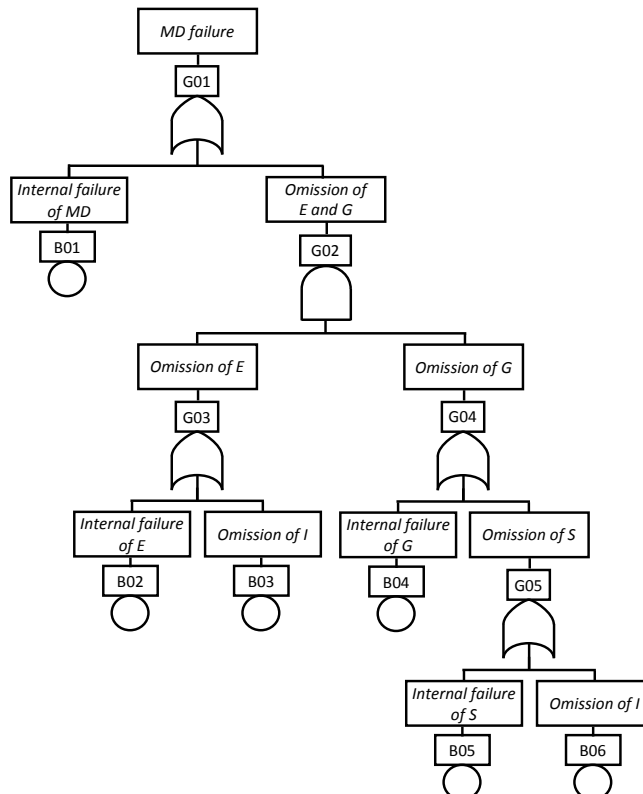


Figure 2.2-5: An abstract model of HPS

2.2.1 Qualitative Analysis

Fault trees contain **cut sets** which are combinations of basic events whose occurrence causes the top event to occur (Vesely *et al.*, 2002). The HPS fault tree in Fig. 2.2-5 is only an equivalent logical graphical representation of the HPS description. From the fault tree, one can tell how HPS works. Qualitative analysis of a fault tree is an evaluation of the fault tree in terms of an equivalent expression for more focused analyses. It involves the determination of cut sets that are necessary and sufficient to cause the top-event or system failure; necessary because each basic event in the combination is needed to trigger the top event and sufficient because the cut sets do not need the occurrence of additional events to cause the top event. These necessary and sufficient combinations of basic events are known as **Minimal Cut Sets (MCS)**. Lee *et al.* (1985) provide a detailed review of techniques used in qualitative analysis; these techniques reduce fault trees into their minimal structure.

Since the focus of this thesis is on quantitative analysis rather than qualitative analysis, we only consider briefly a few techniques (Vesely *et al.*, 2002) for qualitatively analysing fault trees with Boolean algebra – in this case using HPS. The full list of Boolean laws for manipulating Boolean algebra are well known and can be found in Boole (1916). However, some few to remember for the sake of demonstration are:

Distributive Law:

$$A \cdot (B + C) = A \cdot B + A \cdot C \quad (2.2-7)$$

$$A + (B \cdot C) = (A + B) \cdot (A + C) \quad (2.2-8)$$

Idempotent Law

$$A \cdot A = A \quad (2.2-9)$$

$$A + A = A \quad (2.2-10)$$

Identity Law

$$A \cdot I = A \quad (2.2-11)$$

Annulment Law

$$A + I = I \quad (2.2-12)$$

Since the fault tree can be expressed in terms of Boolean logic, Boolean laws can be used to reduce it to its ‘minimal’ logical form. The Boolean distributive laws are used to expand the cut sets whilst the identity and reduction laws are used to eliminate redundancies. This expansion and reduction of cut sets is done repeatedly until MCSs are achieved. The MCSs are usually in a form of a sum of disjoint products.

From the HPS example,

$$MD_f = MD_i + (E_i + I_o) \cdot (G_i + S_i + I_o) \quad (2.2-13)$$

Using the Boolean distributive law (2.2-7) and (2.2-8),

$$\begin{aligned} (E_i + I_o) \cdot (G_i + S_i + I_o) &= E_i \cdot G_i + E_i \cdot S_i + E_i \cdot I_o + I_o \cdot G_i \\ &\quad + I_o \cdot S_i + I_o \cdot I_o \end{aligned} \quad (2.2-14)$$

Substituting (2.2-14) into (2.2-13),

$$MD_f = MD_i + E_i \cdot G_i + E_i \cdot S_i + E_i \cdot I_o + I_o \cdot G_i + I_o \cdot S_i + I_o \cdot I_o \quad (2.2-15)$$

Using (2.2-9) and (2.2-11),

$$I_o \cdot I_o = I_o \cdot 1$$

Using the distributive law again

$$E_i \cdot I_o + I_o \cdot G_i + I_o \cdot S_i + I_o = I_o \cdot (E_i + G_i + S_i + 1) \quad (2.2-16)$$

Finally applying the annulment law (2.2-12) to (2.2-16),

$$MD_f = MD_i + E_i \cdot G_i + E_i \cdot S_i + I_o \quad (2.2-17)$$

Equation (2.2-17) is the list of MCSs leading to the occurrence of the medical device failure. From this expression, it can be seen that the single-event MCSs are lack of an input demand I_o and failure of the medical device itself, MD_i . These single-event MCSs are called critical events because they, by themselves, can cause the top-event to trigger. The remaining MCSs are combinations of basic events and therefore they are not as critical in their contribution to the occurrence of the top event.

These logical relationships are the sort of information qualitative analysis provides to engineers and system designers who work with fault trees of hundreds of events and gates. Based on the results of qualitative analysis, a system can be improved or redesigned to enhance its reliability. For example, it has been seen in the HPS that the medical device is a critical component. Therefore to enhance its reliability a high quality device may be purchased for use in the hospital. Alternatively, if the hospital has enough resources, a standby device may be purchased instead. Again, it can be seen that the electricity system contributes more to the medical device failure than the generator system therefore with limited funds, attention can be focused on getting electric power supply from a more reliable provider and getting a cheaper generator.

2.2.2 Quantitative Analysis

The quantitative analysis of FTs involves the evaluation of its top event probabilities, basic event importance and the uncertainties of these results. The top event quantification determines the unreliability (failure probability) of the top event. The importance quantification determines the contribution or significance of components or cut sets to the occurrence of the top event.

2.2.2.1 Top-Event Probability

In accordance with the laws of probability, all events (top events, intermediate events and basic events) are real numbers, \mathbb{R} , ranging from 0 to 1 inclusively. The probability of an event X occurring at any time before time t is denoted $Pr(X) \{t\}$ and represented by Fig. 2.2-6. This is the same as the CDF, $F(X) \{t\}$, of the event. In the graph, the arc is annotated with its corresponding CDF.

Fig. 2.2-7 below is a timing diagram for Fig. 2.2-6 where '0' is the 'FALSE' or 'not-occurred' state represented by a sunken timeline and '1' is the 'TRUE' or 'occurred' state represented by the raised timeline.

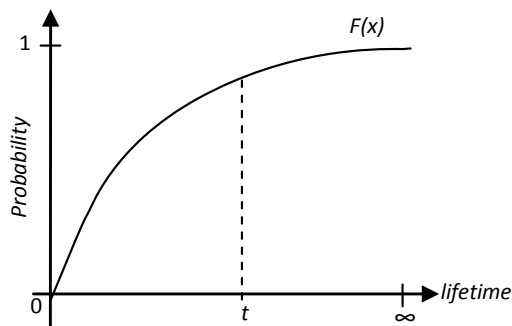


Figure 2.2-6: Graphical representation of an event X

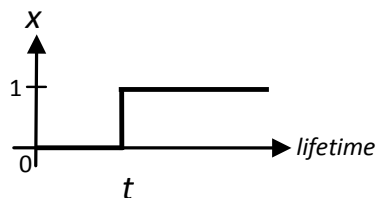


Figure 2.2-7: Timing diagram of an event X

Basic event failure probabilities are necessary for complete quantitative analysis of system to take place. There are various techniques for evaluating traditional fault trees. Again, Lee *et al.* (1985) and Vesely *et al.* (2002) provide a detailed review of quantitative analysis approaches for evaluating fault trees. We classify these approaches into

three main categories in this thesis: Minimal Cut Set (MCS) approach, Binary Decision Diagram (BDD) approach and Monte Carlo (MC) Simulation approach.

From this point onwards, HPS will be used to demonstrate the applicability of the quantitative analysis techniques that are to be reviewed. For this reason, we assume some failure probabilities (given that $t=1$) for HPS' basic events in Table 2.2-2.

Table 2.2-2: Failure Probabilities of HPS' Events

Event	Failure Probabilities (hrs)
MD	0.000562
E	0.000112
S	0.0007562
G	0.0003579
I	0.0000655

Minimal Cut Set approach

This technique is based on the MCS approach described in the qualitative analysis review. Once the MCSs of a fault tree have been determined qualitatively, and basic event failure rates are known, the probability of the top-event occurrence can be calculated.

Given that TE is any top event and n is the number of logically related MCSs of TE , TE can be expressed as Boolean sums of products:

$$TE = MCS_1 + MCS_2 + \dots + MCS_{n-1} + MCS_n$$

Therefore, the probability of TE occurring at a particular time, t is

$$P(TE)\{t\} = \sum_{i=1}^n P(MCS_i)\{t\} \quad (2.2-18)$$

Each MCSs is usually a product term with m independent events, X

$$P(MCS_i)\{t\} = \prod_{k=1}^m P(X_k)\{t\} \quad (2.2-19)$$

There are two major ways of quantitatively evaluating FTs using the MCS approach. These are the Principle of Inclusion-Exclusion (PIE) and the Esary-Proschan (EP) techniques.

According to the Inclusion-Exclusion (Linial & Nisan, 1990; Sherstov, 2009; Gallier, 2010) principle originally developed by Poincare-Sylvester, for any finite sequence of events X_1, \dots, X_n , of $n \geq 2$ subsets of a finite set S ,

$$\left| \bigcup_{k=1}^n X_k \right| = \sum_{\substack{I \subseteq \{1, \dots, n\} \\ I \neq \emptyset}} (-1)^{(|I|-1)} \left| \bigcap_{i \in I} X_i \right| \quad (2.2-20)$$

$$P(TE)\{t\} = P\left(\left| \bigcup_{k=1}^n MCS_k \right|\right) = P_r\left(\sum_{\substack{I \subseteq \{1, \dots, n\} \\ I \neq \emptyset}} (-1)^{(|I|-1)} \left| \bigcap_{i \in I} MCS_i \right|\right) \quad (2.2-21)$$

$$\begin{aligned} Pr(TE)\{t\} &= \sum_{i=1}^n P\{MCS_i\}(t) - \sum_{1 \leq i < j \leq n} P\{MCS_i * MCS_j\}(t) \\ &+ \sum_{1 \leq i < j < k \leq n} P\{MCS_i * MCS_j * MCS_k\}(t) + \dots \\ &+ (-1)^{n-1} P\left\{\prod_{i=1}^n MCS_i\right\}(t) \end{aligned} \quad (2.2-22)$$

Esary and Proschan (1963) proposed a formula for quantifying coherent systems with independent components. This quantification method has been popularised (Henley & Kumamoto, 1981) and used in some software (Item, 2005) thereafter. According to the Esary-Proschan formula, the probability of a system failure (TE) of 'n' minimal cut sets (MCSs) with each cut set having a combination of 'm' basic events is:

$$P(TE)\{t\} = 1 - \prod_{j=1}^n \left(1 - \prod_{i=1}^{m_j} P(MCS_{i,j})\{t\}\right) \quad (2.2-23)$$

Both techniques produce similar results. For example, the top-event probability of HPS using PIE and EP are 6.27587889902248E-4 and 6.27587889902253E-4 respectively. However, as the MCSs grow larger, PIE becomes very computationally expensive. In evaluating HPS, PIE required 31 arithmetic calculations while EP required only 8. For a fault tree with twenty (20) MCSs, PIE will require 10485759 arithmetic calculations while EP will require only 40 calculations. For a fault tree with n MCSs, PIE will require $\sum_{k=0}^n k \times \binom{n}{k} - 1$ arithmetic calculations whilst EP will require only $2 \times n$.

Binary Decision Diagrams (BDD)

A BDD can be used as an alternative approach to analysing classical fault trees. A Binary Decision Diagram (Lee, 1959; Akers, 1978; Bryant, 1986) is technically a directed acyclic graph – a graph with all paths in one direction and it has no loops – with terminal and non-terminal nodes (vertices) connected by branches (edges). Vertices or nodes are represented by circles whilst branches or paths are represented by lines. A BDD has one root vertex. Every other vertex, apart from the root vertex, has only one parent vertex. Each non-terminating vertex has exactly two children.

Unlike the MCS approach, which identifies combinations of basic events necessary and sufficient to cause the top event, a BDD is a graphical representation that encodes the logical structure of a fault tree and is able to perform quantitative analysis via this structure instead of cut sets. The use of BDDs in FTA was developed (Rauzy, 1993; Sinnamon, 1996; Sinnamon & Andrews, 1996) to translate FTs into BDD and then encode the Shannon's decomposition (Schneeweiss, 1989) if-then-else (**ite**) structure.

In FTA, a BDD's non-terminating nodes represent basic events whilst terminating nodes represent the final state of a system or component: 0 for failure or 1 for success. They are represented with boxes in this thesis. Usually, the leftmost path from a vertex leads to a failure and rightmost path leads to a success. They are annotated with a '1' and '0' respectively. Fig. 2.2-8 is a BDD with an **ite** structure for a basic event X . Where, the function $f1$ on the '1' branch is considered if X fails and $f2$ on the '0' branch is considered if X is operational.

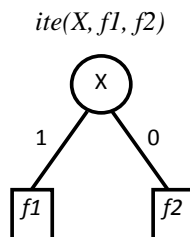


Figure 2.2-8: BDD and ite structure for a basic event X

Due to its efficient data structure encoding, BDDs utilize comparatively low computing resources while producing the same probabilistic and importance measures results as the MCS technique. However, for an efficient BDD analysis to take place, there is need for the events in a fault tree to be appropriately ordered to produce accurate results (Sinnamon & Andrews, 1996). This ordering of events is simply the prioritisation of events; events with higher priority occur higher in the BDD and low priority events oc-

cur lower. In this thesis, $X \rightarrow Y$ means the event X has higher priority over the event Y ; X will have a higher level in the resulting BDD than Y . Also in this thesis, depth-first traversal (top-down, left-right) will be used for BDD analysis unless stated otherwise.

Representing any Boolean operator ‘.’ or ‘+’ with ‘ \diamond ’, the following procedure (Rauzy, 1993; Sinnamon and Andrews, 1998) can be followed for BDD analysis in FTA.

1. Assign an order to the basic events in the fault tree.
2. Generate an **ite** structure for each basic event.
3. For any two event $A = \text{ite}(X, f_{x1}, f_{x0})$, $B = \text{ite}(Y, f_{y1}, f_{y0})$

If $A \rightarrow B$ then,

$$A \diamond B = \text{ite}(X, f_{x1} \diamond B, f_{x0} \diamond B)$$

If $A = B$ then,

$$A \diamond B = \text{ite}(X, f_{x1} \diamond f_{y1}, f_{x0} \diamond f_{y0})$$

Bearing in mind that if \diamond is an OR gate and E is any event,

$$1 \diamond E = 1$$

$$0 \diamond E = E$$

Whilst, if \diamond is an AND gate,

$$1 \diamond E = E$$

$$0 \diamond E = 0$$

We demonstrate the operation of BDD using the HPS case study. To begin with, we consider the order $MD \rightarrow E \rightarrow I \rightarrow G \rightarrow S$.

$$G03 = B02 + B03$$

$$= \text{ite}(E_i, 1, 0) + \text{ite}(I_o, 1, 0)$$

$$= \text{ite}(E_i, 1, \text{ite}(I_o, 1, 0))$$

$$G05 = B05 + B06$$

$$= \text{ite}(S_i, 1, 0) + \text{ite}(I_o, 1, 0)$$

$$= \text{ite}(I_o, 1, \text{ite}(S_i, 1, 0))$$

$$G04 = B04 + G05$$

$$= \text{ite}(G_i, 1, 0) + \text{ite}(I_o, 1, \text{ite}(S_i, 1, 0))$$

$$= \text{ite}(I_o, 1, \text{ite}(G_i, 1, \text{ite}(S_i, 1, 0)))$$

$$G02 = G03 \cdot G04$$

$$= \text{ite}(E_i, 1, \text{ite}(I_o, 1, 0)) \cdot \text{ite}(I_o, 1, \text{ite}(G_i, 1, \text{ite}(S_i, 1, 0)))$$

$$= \text{ite}(E_i, \text{ite}(I_o, 1, \text{ite}(G_i, 1, \text{ite}(S_i, 1, 0))), \text{ite}(I_o, 1, 0))$$

$$\begin{aligned}
\text{Top} &= G0I \\
&= B0I + G02 \\
&= \text{ite}(MD_i, 1, 0) + \text{ite}(E_i, \text{ite}(I_o, 1, \text{ite}(G_i, 1, \text{ite}(S_i, 1, 0))), \text{ite}(I_o, 1, 0)) \\
&= \text{ite}(MD_i, 1, \text{ite}(E_i, \text{ite}(I_o, 1, \text{ite}(G_i, 1, \text{ite}(S_i, 1, 0))), \text{ite}(I_o, 1, 0)))
\end{aligned}$$

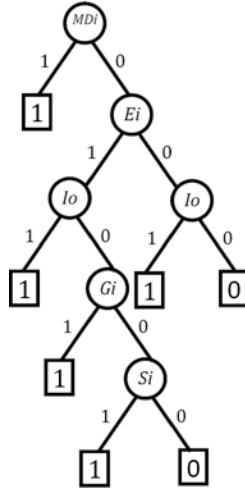


Figure 2.2-9: BDD for HPS using $MD \rightarrow E \rightarrow I \rightarrow G \rightarrow S$

Figure 2.2-9 is a corresponding BDD for the above HPS analysis. Cut sets for HPS can be derived from Figure 2.2-9 by tracing the paths through the BDD from the root vertex to a terminal vertex with a “1” value. Only basic events on branches labelled “1” are considered when tracing these terminal vertices. Therefore, the cut sets for the above BDD are:

1. MD_i
2. $E_i . I_o$
3. $E_i . G_i$
4. $E_i . S_i$
5. I_o

It is evident that the cut sets listed above are not minimal; I_o is a critical event and so redundant in $E_i . I_o$. The power of BDD is in its ability to calculate the top event probability from a fault tree directly without having to minimise it (i.e., generate MCS for it). To evaluate the top-event probability of a fault tree using BDD, the probability of the sum of logically disjoint paths in the BDD is calculated (Sinnamon and Andrews, 1996). Logically disjoint paths are paths that lie on branches labelled “0”– branches representing event failures. We represent events lying on logical disjoint paths with a bar on top of them, for example, \bar{X} , which means NOT X . \bar{X} logically means X has not occurred and probabilistically is $1 - P(X)$, which is the reliability of X .

Disjoint paths through Figure 2.2-9 are:

1. MD_i
2. $\overline{MD}_i \cdot E_i \cdot I_o$
3. $\overline{MD}_i \cdot E_i \cdot \overline{I}_o \cdot G_i$
4. $\overline{MD}_i \cdot E_i \cdot \overline{I}_o \cdot \overline{G}_i \cdot S_i$
5. $\overline{MD}_i \cdot \overline{E}_i \cdot I_o$

The top event is calculated by finding the probability of the sum of disjoint paths (Rauzy, 1993). Summing all the disjoint paths obtained from HPS together produces 6.27587859612539E-4.

To demonstrate the impact of using different ordering in BDDs, we use a different order $S \rightarrow MD \rightarrow I \rightarrow E \rightarrow G$ to analyse HPS.

$$\begin{aligned} G03 &= B02 + B03 \\ &= \text{ite}(E_i, 1, 0) + \text{ite}(I_o, 1, 0) \\ &= \text{ite}(I_o, 1, \text{ite}(E_i, 1, 0)) \end{aligned}$$

$$\begin{aligned} G05 &= B05 + B06 \\ &= \text{ite}(S_i, 1, 0) + \text{ite}(I_o, 1, 0) \\ &= \text{ite}(S_i, 1, \text{ite}(I_o, 1, 0)) \end{aligned}$$

$$\begin{aligned} G04 &= B04 + G05 \\ &= \text{ite}(G_i, 1, 0) + \text{ite}(S_i, 1, \text{ite}(I_o, 1, 0)) \\ &= \text{ite}(S_i, 1, \text{ite}(I_o, 1, \text{ite}(G_i, 1, 0))) \end{aligned}$$

$$\begin{aligned} G02 &= G03 \cdot G04 \\ &= \text{ite}(I_o, 1, \text{ite}(E_i, 1, 0)) \cdot \text{ite}(S_i, 1, \text{ite}(I_o, 1, \text{ite}(G_i, 1, 0))) \\ &= \text{ite}(S_i, \text{ite}(I_o, 1, \text{ite}(E_i, 1, 0))), \text{ite}(I_o, 1, \text{ite}(E_i, \text{ite}(G_i, 1, 0), 0))) \end{aligned}$$

$$\begin{aligned} \text{Top} &= G01 \\ &= B01 + G02 \\ &= \text{ite}(MD_i, 1, 0) + \text{ite}(S_i, \text{ite}(I_o, 1, \text{ite}(E_i, 1, 0))), \text{ite}(I_o, 1, \text{ite}(E_i, \text{ite}(G_i, 1, 0), 0))) \\ &= \text{ite}(S_i, \text{ite}(MD_i, 1, \text{ite}(I_o, 1, \text{ite}(E_i, 1, 0))), \text{ite}(MD_i, 1, \text{ite}(I_o, 1, \text{ite}(E_i, \text{ite}(G_i, 1, 0), 0)))) \end{aligned}$$

The above BDD expression for the top-event produces the BDD diagram in Figure 2.2-10. It is evident that the cut sets are $S_i \cdot MD_i$, $S_i \cdot I_o$, $S_i \cdot E_i$, MD_i , I_o and $E_i \cdot G_i$.

The disjoint paths through Figure 2.2-10 are:

1. $S_i \cdot MD_i$
2. $S_i \cdot \overline{MD}_i \cdot I_o$
3. $S_i \cdot \overline{MD}_i \cdot \overline{I}_o \cdot E_i$
4. $\overline{S}_i \cdot MD_i$
5. $\overline{S}_i \cdot \overline{MD}_i \cdot I_o$
6. $\overline{S}_i \cdot \overline{MD}_i \cdot \overline{I}_o \cdot E_i \cdot G_i$

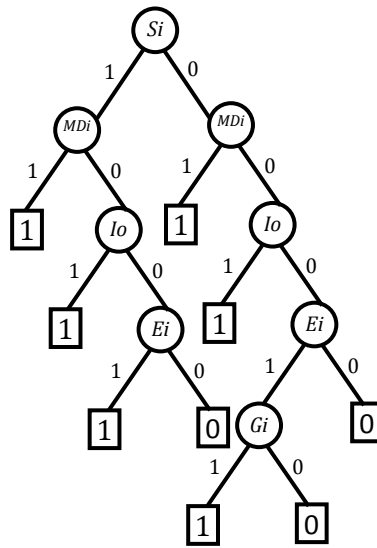


Figure 2.2-10: BDD for HPS using $S \rightarrow MD \rightarrow I \rightarrow E \rightarrow G$

Using the failure data in Table 2.2-2, the top-event of the BDD in Figure 2.2-10 is calculated by summing the disjoint products listed above which results in the value $6.27587859612539E-4$ – the same as the previous BDD result – the only difference being the number of cut sets generated.

Monte Carlo (MC) Simulation

A simulation is a means of learning something about the real world by imitating a scenario using a model. Simulations provide myriads of benefits. They are used in situations where real world scenarios are financially costly, safety hazards, complicated to design or time consuming to implement (Clark and Daigle, 1997; Rozenblit, 2001; Pasquale, 2010). Monte Carlo (MC) simulation is a popular mathematical simulation technique used in various fields such as chemistry, engineering, medicine, games, finance, and telecommunications. It provides numerical solutions to complex problems that are difficult to solve analytically by generating suitable random numbers and ob-

servicing that fraction of the numbers which obey some properties (Weisstein, 2013). Its use has also been extended into reliability analysis (Dutuit and Rauzy, 1997; Wang and Pham, 1997; Marseguerra *et al.*, 1998; Bremaud, 1999; Ejlali and Ghassem Miremadi, 2004; Rocco and Muselli, 2004; Zio *et al.*, 2007; Durga Rao *et al.*, 2009; Manno *et al.*, 2012).

The estimation θ (Terejanu, 2013) for N trials with a standard Monte Carlo method, where $\psi(x_i)$ is a generated random number obeying some properties, is defined as

$$\theta_N = \frac{1}{N} \sum_{i=1}^N \psi(x_i) \quad (2.2-24)$$

The MC method is entirely reliant on the use of random numbers for the computation of its results. The general method involved in creating a MC model includes defining the probability distribution of variables; calculating the cumulative probability distribution for each of these models, generating random numbers, and finally simulating a series of trials. Once the model is created, results can be evaluated after the simulation.

A typical MC simulation begins with modelling the system under study. Once this is done, the model is simulated or ‘run’ by generating random numbers for the model variables to create a unique ‘instance’ of the model. The system variables are generated several times, called trials, to create several instances of the model. These instances are examined for some common predetermined property (Pukite and Pukite, 1998; Bremaud, 1999), which eventually determines the behaviour of the model. There are various types of Monte Carlo methods (Weinzierl, 2000). In this thesis we focus on Monte Carlo Integration.

For example, to evaluate the probability of $C = A \cdot B$, that is $Pr(C) = Pr(A \cdot B)$, using MC simulation, the first thing to do is to model the system. For this model to be constructed, the following conditions must be considered:

- Evaluating the probability of C requires the probabilities of A and B .
- Evaluating the individual probabilities of A and B requires that their random numbers a_i and b_i are less or equal to the probability of A and B respectively (Yevkin, 2010).

With these conditions in mind, the following steps can be followed to evaluate C :

1. Set a variable for summation, S , to zero.
2. Generate two random numbers to simulate an A probability, a_i , and a B probability, b_i

3. If a_i is less than or equal to $P(A)$ AND b_i is less or equal to $P(B)$ then S is incremented by 1.
4. Steps 2 and 3 are repeated a large number of times, T .
5. $P(C)$ is the ratio of S to T .

Algorithms 2.2-1 and 2.2-2 are generic Monte Carlo simulations for the OR and AND gates with n events respectively. $X[n]$ is an array of the failure rates of events. S is a variable to sum the number of instances possessing the AND/OR behaviour. F is a flag and T is the number of trials. $R[i]$ is a random number corresponding to the failure probability, $P(X[i])\{t\}$, of $X[i]$.

Algorithm 2.2-1: A Generic Monte Carlo simulation for the OR gate

```

Require: X[n]
S ← 0
F ← false
for k = 1 to T do
  for i = 1 to n do
    R[i] ← NextRandomNumber
    if (R[i] ≤ Pr(X[i])\{t\}) then
      F ← true
    end if
  end for
  if (F) then
    S ← S + 1
    F ← false
  end if
end for
return S/T

```

Algorithm 2.2-2: A Generic Monte Carlo simulation for the AND gate

```

Require: X[n]
S ← 0
F ← true
for k = 1 to T do
  for i = 1 to n do
    R[i] ← NextRandomNumber
    if (R[i] > Pr(X[i])\{t\}) then
      F ← false
    end if
  end for
  if (F) then
    S ← S + 1
    F ← true
  end if
end for
return S/T

```

As described earlier, the Monte Carlo simulation of a system is done by generally simulating various instances of the system and examining these instances for some common behaviour. Algorithm 2.2-3 is a Monte Carlo simulation for the quantification of HPS.

Algorithm 2.2-3: Monte Carlo simulation for quantifying HPS

Require: E, I, S, G, MD

$S \leftarrow 0$

for $k = 0$ **to** T **do**

$rndE \leftarrow \text{NextRandomNumber}$

$rndI \leftarrow \text{NextRandomNumber}$

$rndS \leftarrow \text{NextRandomNumber}$

$rndG \leftarrow \text{NextRandomNumber}$

$rndMD \leftarrow \text{NextRandomNumber}$

if ($rndMD \leq MD$ **OR** ($rndE \leq E$ **AND** $rndG \leq G$) **OR** ($rndE \leq E$ **AND** $rndS \leq S$)
 OR $rndI \leq I$) **then**

$S \leftarrow S + 1$

end if

end for

return S/T

In Algorithm 2.2-3, random number are generated for each event, E, I, S, G and MD in the HPS MCS. If all MCS behaviours in the 'if' statement have been met the counter S is incremented. This represents an instance of the simulation known as a trial. The entire process is done for T trials. Running the simulation 5000000 times using the failure data in Table 2.2-2 produces a value of 6.282E-4. Figure 2.2-11 is a graph showing how the number of trials affects the accuracy of results. It is evident that the results converge towards the value provided by BDD and MCS approaches.

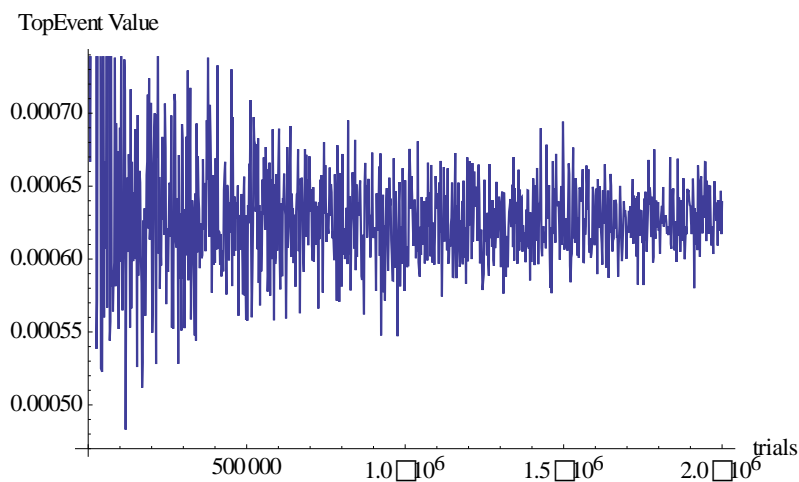


Figure 2.2-11: Graph for Monte Carlo simulation of HPS

The number of trials used in the above case study is large because the values of failure data used are relatively small and will be very scarcely generated by a standard random number generator. For this reason, a large sample space is needed for accurate results to be produced. The 95% lower and upper confidence levels for the above simulation using a linear regression are 6.40876E-4 and 6.54627E-4 respectively with a best fit (linear) of 6.47751E-4.

MC simulation has also seen various improvements to improve its computational efficiency; (Manno *et al.*, 2012) and (Wang and Pham, 1997) provide brief and detailed reviews of MC in reliability engineering respectively. Yevkin (2010) is a recent improvement that increases the efficiency of MC analysis of non-repairable FTs with sequential dependencies existing between their events. After a comparative analysis of various techniques, Yevkin (2010) asserts that “the most significant [MC improvement technique] is the importance sampling approach to reduce standard error of the mean ... [and] is valid for highly reliable systems”.

More recently, Meedeniya *et al.* (2011) and Trubiani *et al.* (2013) have provided a MC simulation based on the dynamic stopping technique to reduce the number of MC samples – improving its computational efficiency. In their technique, the simulation is regulated with respect to some preset conditions; the simulation is executed until these conditions are satisfied. Therefore, a large number of trials may not be necessary for simulation as with the standard Monte Carlo method. A comparative analysis of this technique and the standard Monte Carlo technique is provided in chapter six.

2.2.2.2 Importance Measures

Another significant quantitative measure of FTA is the top event importance which measures the sensitivity of the top-event to basic or intermediate events. Importance measures determine the various contributions of basic or intermediate events to the occurrence of the top event or how a change in any of these events can affect the occurrence of the top event.

Component importance evaluations do not only provide valuable information about their contribution to the top event occurrence, they serve as useful source of data for resource allocation (upgrade, quality, maintenance and the likes). This information aids stakeholders in improving a system (safety, reliability, efficiency, effectiveness etc.) whilst reducing its financial impact.

The worth or ration of a safety-critical system achieving its present level of risk is known as the **Risk Achievement Worth** (RAW) whilst the worth of such a system further reducing its risk is the **Risk Reduction Worth** (RRW). Technically RAW is “the increase in risk [of a system] if the feature [component or event] were assumed not to be there or to be failed” and RRW is “the decrease in risk [of a system] if the feature [basic event] were assumed to be optimised or were assumed to be made perfectly reliable” (Vesely *et al.*, 1986).

Detailed reviews of these importance quantities are available in (Vesely *et al.*, 1986; van der Borst and Schoonakker, 2001; Vesely *et al.*, 2002). However, in this thesis, we discuss two of these quantities: Fussell-Vesely importance and Birnbaum importance.

The **Fussell-Vesely Importance** (Vesely *et al.*, 2002), also known as the Top Contribution Importance, is the contribution of a particular basic event to the top event occurrence probability given that the system has failed. This is calculated by summing all the MCSQs containing a particular event. For any system, S , the FV importance (Henley and Kumamoto, 1981) of an event, X , is given as:

$$FV_X = \frac{P(\sum_i^n MCSQ_i)\{t\}}{Q(S)\{t\}} \quad (2.2-25)$$

Where n is the number of MCSQs containing X .

The second importance measure is the Birnbaum Importance Measure (BM). BM measure determines the sensitivity of the top event probability with respect to some given events. Meaning, BM measures the rate of change in the top event probability with respect to the changes of the probability of a specific event. For a system, S , the BM importance (Henley and Kumamoto, 1981) of an event, X , is given as:

$$BM_X = \frac{\partial Q(S)\{t\}}{\partial Q(X)\{t\}} \quad (2.2-26)$$

2.2.3 Limitations of FTA

Modern safety-critical systems are complex and dynamic. To evaluate such systems appropriately, one will have to capture the sequential dependencies between their components. Failure to do so can result in the underestimation or overestimation of qualitative and/or quantitative results. When a system's reliability is underestimated, it is assumed that the system is less reliable. Therefore, the system is improved to attain a higher level of reliability which may be unnecessary because the system was wrongly evaluated – underestimation. Overestimation is the more dangerous of the two. When a safety-critical system's reliability is overestimated, the actual reliability of the system is lower than the estimated value. This can be very dangerous when the overestimated value is excessively higher than the actual system reliability value. For these reasons, it is expedient that a safety-critical system's reliability is appropriately evaluated.

One of traditional FTA's major limitations is its inability to consider the sequential dependencies between component failures. This means it is unable to capture the dynamic behaviours present in modern systems. For this reason, it is referred to as **static** (Gulati and Dugan, 1997; Merle and Roussel, 2007); traditional FTAs are also known as Static FTAs (SFTA). This is due to the fact SFTs use only the Boolean gates AND and OR which are inadequate to represent the failure behaviour of systems with particular sequence-dependent components. This significant disadvantage can lead to the inaccurate evaluation of a system's reliability (Dugan *et al.*, 1992; Dugan and Doyle, 1997; Walker and Papadopoulos, 2008).

To explain STFA's limitation in more detail, let us go back to the HPS case study. According to the MCSs produced by SFTA, the failure of the electricity, *E*, sub-system and sensor-subsystem, *S*, will result in a failure of the medical device, *MD*. Let us introduce some dynamic behaviour into the MCS: *E* failing *before* *S* or *S* failing *before* *E*. In the situation where *E* fails before *S*, the system does not necessarily fail. The reason being, immediately *E* fails, *S* (at this point *S* has not failed) detects an omission from *E* and activates the standby generator, *G*, to provide power to *MD*. On the contrary, if *S* fails before *E*, the entire system fails. This is so because, when *E* fails, *S* has already (dormant) failed and is unable to activate the generator therefore there will be no power supply to *MD*. Therefore, listing $E \cdot S$ as a MCS leading to the occurrence of *MD* failure is an extremely pessimistic result, which can be costly. There are situations where SFTA results are also too pessimistic (Walker 2009).

The solution to SFTA's limitation can be considered in three broad areas:

1. It must have the ability to capture and model dynamic behaviours. To do this, there is the need for the introduction of novel logical gates (in addition to the classical AND and OR gates), which will be able to capture the temporal behaviours that exist between the components of a dynamic systems. For example, it must be able to model the possible *before* relation that can exist between two events. A typical case is evident in the HPS model, where *S before E* can result in the total failure of *MD*.
2. It must be able to perform qualitative analysis where necessary. Boolean gates have underpinning laws for evaluating Boolean algebra when evaluating classical fault trees. There is also a necessity for the logic relating dynamic gates to have a set of laws, which will serve as a catalyst for reducing dynamic fault trees

into their minimal forms. This will require the firm understanding of the logical dependencies between dynamic components.

3. It must be viable for probabilistic analysis. Any dynamic gate prescribed to model a dynamic behaviour must be suitable for probabilistic evaluation. The probabilistic evaluation of these gates in relation to others (dynamic and Boolean) must also be determined. This will enable the probabilistic evaluation of the top-event.

Unfortunately, all the techniques discussed for the quantitative analyses of SFTs are also limited and unable to perform quantitative analysis of dynamic systems. The MCS and Monte Carlo techniques depend on MCS obtained from qualitative analysis. Therefore, since the MCS is unable to capture the dynamic behaviour of systems, both techniques are unable, at this level, to include dynamic behaviours in their analyses. BDD, the third approach, though it does not depend on MCS, has no theoretical or practical structure to incorporate dynamic behaviours in its analysis. In conclusion, all these techniques for quantitatively analysing classical fault trees are inadequate for analysing modern dynamic safety-critical systems because they cannot capture their dynamic behaviours.

Extended FTA (Vesely *et al.* 2002) is one of the earliest improvements made to classical FTA. It introduces some FT modelling principles. These include common cause failure (CCF), human error and loop and feedback. It also alters some naming schemes and ground rules stated in the classical FTA definition. However, these changes do not make classical fault tree analysis formidable enough to tackle challenges modern systems present.

Other earlier (pre 1990s) improvements to FTA were heavily centred on the efficiency of the analysis (both qualitative and quantitative) of large fault trees, evaluation of importance measures and the use of simulation. The only major publication made on the analysis of dynamic gates in FTA is on the quantitative analysis of Priority-AND (PAND) gates (Fussell *et al.*, 1976). This paper describes the PAND scenarios and provides a model for it. It also derives a mathematical expression for its quantification of n events. However, it makes no mention of qualitative analysis. Though it seemed that FTA's fate in analysing dynamic systems was forgotten, there was a massive breakthrough in the early 1990s. During this time, FTA saw a revolutionary turn towards the dynamic behaviours of modern systems when the Dynamic Fault Tree (DFT) analysis was invented. This caused an explosion of investigation in this field of research.

2.3 Dynamic Fault Trees (DFT)

DFT (Dugan *et al.*, 1992) was proposed to extend SFT and allow analysts to tackle computer-based systems, which feature dynamic behaviours and require the accurate evaluation of their outcomes – the major limitation of SFT. To do this, DFT introduced special gates to model possible behaviours existing between dynamically related components. These gates, known as dynamic gates, are the Functional Dependency (FDEP) Gate, Cold spare Gate, Priority AND (PAND) Gate, and the Sequence-Enforcing Gate (Dugan *et al.*, 1990; Dugan *et al.*, 1992; Vesely *et al.*, 2002). DFT uses these dynamic gates in addition to the static gates of SFTs in its analysis. To comprehend the structural and logical functions of these gates in detail, we consider them one after the other.

Priority-AND (PAND) Gate

As earlier discussed, the PAND gate is one of the oldest dynamic gates described and analytically analysed in the late 1970s by Fussell *et al.* (1976) to model particular sequences in which basic events fail. It was defined as being “logically equivalent to an AND gate where the input events must occur in a specific order” (Fussell *et al.* 1976). According to the Fussell *et al.*, the output event of a PAND gate is triggered when all its input events occur and they do so in a particular sequence – one occurs after another. DFT’s PAND is the same as that of Fussell *et al.* However, the only difference between the two is the structural/graphical representation of the gate with n events. While the former represents a PAND gate with n events with cascading gates of two events (depth wise), the later represents it with one single gate having all its inputs underneath it (breadth wise). Both representations have the same interpretation and quantitative result.

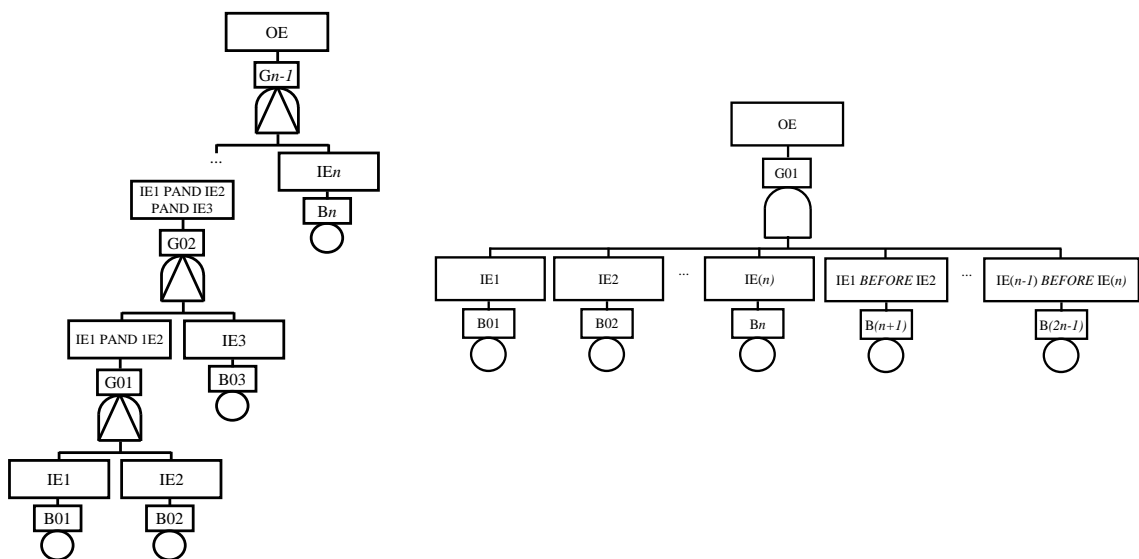


Figure 2.3-1: A (right): PAND gate for n events, B (left): equivalent SFT

Figure 2.3-1A (right) is a representation (depth wise) of the PAND gate with n events. In the diagram, the initial input events (IE1 and IE2) are fed into the first PAND gate, G01. G01 and the subsequent input event, IE3, serve as input events to G02. G02 and the subsequent event also are used as input events to the next gate; on and on this is repeated until the last PAND gate and last input event are reached. Figure 2.3-1B on the left is a representation of the PAND gate with an AND gate. This representations contains the *BEFORE* expression to represent the dynamic behaviour missing in the classical AND representation.

Cold-Spare (CSP) Gate

A redundancy, standby or spare component is a component that replaces a primary component when the primary component fails. A spare gate (Dugan *et al.*, 1990; Dugan *et al.*, 1992) is used to model the situation where one or more spare components can replace a primary component when it fails. It takes in one or more basic inputs events and has one output that becomes true after all its input events have occurred. The left-most input event is usually the primary component and the subsequent events are the standby components. During the initial operation of the system, the primary input event is usually active (turned on) while the spare input events are inactive (turned off), waiting to replace the primary component when it fails. When the primary component fails, the subsequent spares are activated sequentially; leftmost to rightmost. Depending on its dormancy factor, spare gates could be termed as cold, hot or warm (Dugan *et al.*, 1992). A spare gate is cold if it does not fail before it is activated – dormancy factor is zero. A spare gate is hot when its failure behaviour as a spare is the same as while it is active – dormancy factor is one. Finally, a spare gate is warm if its failure behaviour is between hot spare and cold spares. Figure 2.3-2 is a graphical representation of the cold spare gate with n input events where *PC* is the primary component, *SC* the spare component and *OE* the output event.

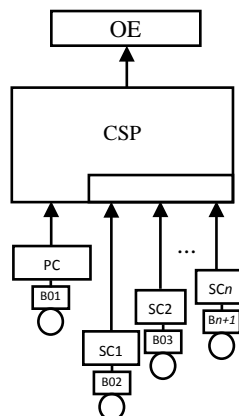


Figure 2.3-2: A cold spare gate with n events

A special case of spare gates worth noting is their ability to model spare components shared between two or more primary components; shared spare input events feed more than one spare gate as shown in Figure 2.3-3.

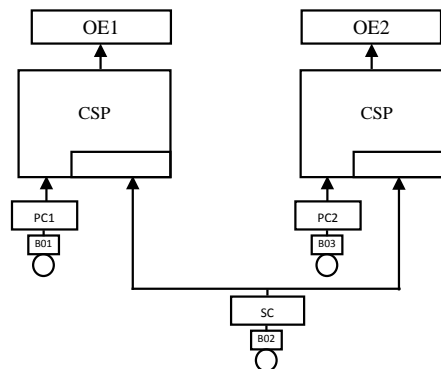


Figure 2.3-3: A spare component for two primary components

In Figure 2.5-3, *SC* is a spare component for primary components *PC1* and *PC2*. *SC* can replace only one failed component at a time – usually the first component to fail. Therefore, if *PC2* fails first, *SC* is activated to replace *PC2*. This makes *SC* unavailable so if *PC1* fails afterwards, there will be no spare component to replace it and *OE1* occurs; *OE2* will occur if *SC* fails.

Functional Dependency (FDEP) Gate

The FDEP gate (Dugan *et al.*, 1990; Dugan *et al.*, 1992) was introduced to model common cause failures (CCF) of events. A CCF occurs when the occurrence of a single event failure triggers the failure occurrence of other dependent events almost at the same time. A real world example is the Browns Ferry Nuclear Power Plant fire incident in 1975 (USNRC, 2013) where “cables for power, control systems and instrumentation” of a nuclear reactor plant were ‘simultaneously’ damaged by fire. The FDEP gate has three basic components (Dugan *et al.*, 1992):

- a) a ‘trigger-input’ which is a basic event or the output of another gate.
- b) a ‘non-dependent output’ which reflects the status of the trigger event
- c) one or more ‘dependent basic events’ which are functionally dependent on the trigger events.

The dependent basic events are activated when the trigger event fires. However, the individual occurrence of any dependent basic event has no influence on the trigger event. Figure 2.3-4 (Vesely *et al.*, 2002) is a graphical representation of an FDEP gate with a trigger event *TE* and *n* dependents events *DEs*. The non-dependent outputs are connect-

ed to the fault trees with dashed lines; they do not have any logical output (Vesely *et al.*, 2002).

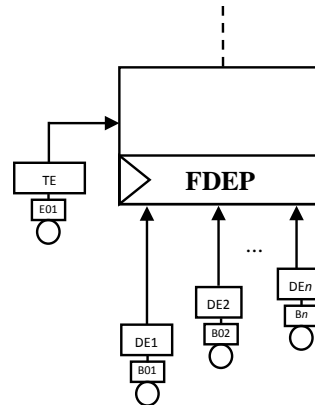


Figure 2.3-4: An FDEP gate with a trigger and n dependent events

Sequence Enforcing (SEQ) Gate

A Sequence-Enforcing (SEQ) gate “allows events to occur in a particular order” unlike the PAND gate which “detects whether events occur in a particular order” (Dugan *et al.*, 1992). Meaning the SEQ gate enforces or constraints a set of events to occur in a specific sequence; the output event occurs if all input events occur in the specific sequence. It has an output event and a list of n input events.

2.3.1 DFT Qualitative Analysis

As mentioned earlier, traditional Boolean laws are incapable of analysing dynamic fault trees. For qualitative analyses of DFTs to take place, there must be underpinning laws for representing and manipulating the behaviour of the new DFT logic gates and their relationships with one another. DFT, since its inception, has been used mainly as a quantitative tool to calculate the probability of a system failure occurring.

Relatively little effort has been expended on qualitatively analysing DFTs. One of the earliest works on the qualitatively analysing DFTs was done by Tang and Dugan (2004) which introduced ‘*minimal cut sequences*’ – a set of ordered events. DFTs are usually analysed using Markov Chains. As popularly known, Markov chains can become very cumbersome and time-consuming to analyse as the chains grow bigger. Tang and Dugan (2004) acknowledged that extracting temporal information from Markov chains for further analysis would also be time-consuming so they proposed an approach that is based on BDDs. This approach commences with the decomposition of dynamic gate constraints into logical constraints and timing constraints which represent the logical relations (AND and OR) and the sequential relations respectively. For example, a PAND gate will be decomposed into ‘AND’ logic constraint and ‘must fail in order

from left-to-right' timing constraint. After this, dynamic gates are replaced with their corresponding static logical constraints and a Zero-suppressed BDD (ZBDD) is derived from the resulting tree and minimised to produce MCSs. Finally, the resulting MCSs are expanded using the timing constraints to produce minimal cut sequences. Given the use of BDDs, this approach should perform better than its Markov chain counterpart. However, the drawback of this approach is that, since temporal dependencies are not considered during the minimisation stage, it is possible that the resulting minimal cut sequences will contain some redundancies or contradictions. This, unfortunately, can lead to inaccurate quantitative results.

Liu *et al.* (2007) proposed non-Markov Chains DFT based algorithm – Cut Sequence Set Algorithm (CSSA) – for generating a Cut Sequence Set (CSS), which is a set of event failures with sequences. CSSA prescribes a temporal failure relationship between any two events and represents cut sequences with what they called sequential failure expression (SFE). After this, a set of transformations are done to translate the SFE into CSS. Liu, Zhang, *et al.* (2007) provide technique to quantitatively analyse the CSS.

Merle (2010) provides enhancements to DFT both quantitatively and qualitatively. He provides an algebraic model and structure function for all dynamic gates in DFTs, as well as laws and their proofs to serve as a formidable theoretical foundation for the full qualitative and quantitative analysis of DFTs, just as done for SFTs.

A more recent advancement in the qualitative analysis of DFTs is provided by Yi *et al.* (2013). The authors provide temporal rules and proofs for the qualitative analysis of DFTs. Unfortunately, these rules are less comprehensive than that provided by Merle (2010). However, the authors claim that their work is to provide a 'completeness' of the techniques proposed by Liu *et al.* (2007) and Merle (2010). By 'completeness' they mean reducing DFTs to cut sequences.

2.3.2 DFT Quantitative Analysis

DFT, since its inception, has become a very popular technique in the field of reliability engineering for quantitative analysis. DFT's quantification can be considered under two main categories: analytical and simulation. The analytical techniques are usually deterministic techniques that are based on Markov Chains, Poisson process, Bayesian networks and Petri nets whilst the simulative approaches are usually estimation methods based on Monte Carlo simulation. The analytical techniques can also be sub-classified

into combinatorial or state-space models (Chiacchio *et al.* 2011). The combinatorial methods are used for evaluating static gates whilst the state-space approaches are used for dynamic gates.

Techniques for evaluating combinatorial methods (MCS, BDD Monte Carlo) have been discussed in the previous section. There are various state-space approaches for DFTs. Chiacchio *et al.* (2011) and (Merle 2010) provide some detailed review of some of these techniques. In this thesis, we provide a focused review by considering the major state-space and simulation approaches.

Continuous Time Markov Chains (CTMC)

Markov models are widely used for analysing continuous time, discrete state scenarios (DoD, 1998; Pukite and Pukite, 1998) in reliability engineering. DFTs can be quantified by converting dynamic fault trees into CTMC (Dugan *et al.*, 1992). DFT analysis commences by converting the fault tree with dynamic gates into state models. These models are modularised (Gulati and Dugan, 1997) into static and dynamic modules; static modules contain only static gates while dynamic modules contain dynamic gates. Static sub trees are evaluated using Zero-suppressed BDD-based approaches (Gulati and Dugan, 1997; Tang and Dugan, 2004) while the dynamic sub trees are evaluated using CTMC. Once the quantification of both static and dynamic sub trees is done, the modules are represented as basic components. The resulting top-level FT is then evaluated as a static tree with many basic events. Using a BDD approach to quantify SFTs is very fast but the use of CTMCs in analysing DFTs tends to suffer state-space explosion when the DFT gets large.

Apart from the state-space explosion problem, Markov-based approaches are difficult to model and analyse for large fault trees and they are restricted to exponential distributions only (Dugan *et al.*, 1990; Boudali and Dugan, 2006).

Bayesian Networks (BN)

Bayesian Networks (Pearl, 1985) or probabilistic dependence graphs have been a probabilistic technique used in uncertainty analysis (Bobbio *et al.*, 2001) over the past two decades. Bobbio *et al.* (2001a) introduced the application of BN into FTA. In their approach, they mapped classical fault trees with only AND and OR gates to Bayesian networks and quantitatively analysed this resulting Bayesian network using two practical methods: a forward/predictive method and a backward/diagnostic method. A similar

approach of mapping classical FTA in the form of Reliability Block Diagrams (RBD) to BN has been exploited (Torres-Toledano and Sucar, 1998). However, these BN approaches are also restricted by the inability of Boolean gate not being able to capture the sequential dependencies between components.

Improvements to include dynamic/temporal features into BN for reliability analysis include DBN (Weber and Jouffe, 2003; Montani *et al.*, 2005; Montani and Portinale, 2006; Salem *et al.*, 2006), DTBN (Boudali and Dugan, 2005), CTBN (Boudali and Dugan, 2006) and more recently (Marquez *et al.*, 2008). Unfortunately, all the above BN approaches provide only quantitative analysis (Merle, 2010).

Petri Nets (PN)

Petri Nets (Peterson, 1977) are alternative graphical modelling structures for representing and evaluating fault trees (Bobbio and Franceschinis, 2003). They have arcs that connect nodes to transitions or vice versa – similar to state machines. Reviews of the application and techniques based on PN for reliability analysis are provided in Aldemir *et al.* (2006) and Sadou and Demmou (2009). Some well-known PN-based techniques include GSPN (Marsan, 1989; Malhotra and Trivedi, 1995), CSPN (Sknourilova and Bris, 2008). Unlike CTMCs where DFTs are converted into failure automata, DFTs can be converted into various derivatives of PN (such as General SPN) before being translated into a CTMC for quantitative analysis. The main advantage of most PN-based techniques is their ability to evaluate a system using Markov chains (Aldemir *et al.*, 2006). Unfortunately, PN-based techniques inherit some of Markov chain based technique's limitations: they are prone to the state-space explosion (Sadou and Demmou, 2009) and they consider only the exponential distribution of components and are solely used for quantitative analysis (Merle, 2010). Sadou and Demmou (2009) propose a technique to minimize the state-space explosion problem of PN by extracting certain aspects of the graph susceptible to explosion and analysing these sub graphs differently.

Monte Carlo Simulation

In an earlier section, the discussion of how MCs can be used in quantitatively analysing SFTs was made. However, when it comes to DFTs, some sequential ordering issues must be tackled. Recent research by Rao *et al.* (2009) has succeeded in implementing the dynamic gates (PAND, SEQ, SPARE and FDEP) with the MC simulation technique. This led to the development of a tool called DRSIM (Rao *et al.*, 2009) for quantitative analysis of DFTs using MC simulation. Just like most MC simulation tools, the major

drawback of DRSIM is the computational cost of running MC; however, the developers think with the incredible development of computing power, this should not be a major challenge. Unfortunately, the quantitative computation of large fault trees present in the real world hybrid systems where rare events exist continue to pose computational challenges (Villen-altamirano and Villen-altamirano, 1994).

Algebraic Model

Amari *et al.* (2003) proposed algebraic model for evaluating each gate in DFTs but make no mention of how qualitative analysis of cut sequences is performed. A more recent DFT quantitative technique is the algebraic framework for specifying the structure function of DFTs (Merle, 2010; Merle *et al.*, 2013). The algebraic approach is formulated from first-principle and does not depend on the failure distributions of system components. It prescribes temporal semantics for the analysis of the Boolean operators AND and OR and temporal operators BEFORE (BF) and SIMULTANEOUS (SM) (Merle and Roussel, 2007). It then provides algebraic expressions for the structure functions of DFTs with all the dynamic gates: PAND (Merle and Roussel, 2007; Merle *et al.*, 2010), FDEP (Merle *et al.*, 2009) and Spare (Merle *et al.*, 2010). Once these algebraic expressions have been produced, cut sequences can be determined and minimised to produce the sum-of-products, which are used to evaluate the top event probability. Merle *et al.* (2013) provide a summary of the algebraic framework of DFT and apply it to a case study.

2.3.3 Limitations of DFT

Inclusion of temporal behaviour into static structures does not come cheap. Several factors have to be considered and handled appropriately. The introduction of DFT solved SFT's major limitation by introduction novel gates that modelled the dynamic behaviours in modern safety-critical systems. Unfortunately, DFT's major limitation was its use of Markov Chains for quantitatively analysing fault trees which by-passed qualitative analysis. It is a known fact that the complexity of Markov chains explodes exponentially out of proportion when the chains grow larger. This makes it very cumbersome and difficult to generate. Fortunately, various improvements have been made to solve the state explosion problem by the use of modularisation, Monte Carlo simulation, algebraic modelling among others. Attempts have also been made to qualitatively analyse DFTs. In spite of these enhancements, some fundamental issues need resolving.

The major drawback of the DFT technique is the ambiguity in the definitions of its gates – PAND, FDEP and SPARE. These ambiguities arise because DFTs do not consider the simultaneous occurrences of events. Details of these issues are discussed in (Coppit *et al.*, 2000) and (Walker, 2009). We review one such ambiguity.

DFT adopts (Dugan *et al.*, 1992) the PAND gate described by Fussell *et al.* (1976). This PAND, as described earlier, occurs if the input events occur in a particular sequence – one before another or at the same time – therefore, it is referred to as the inclusive PAND because it includes the simultaneous occurrence of events. Technically, $A < B = A \text{ BEFORE } B \text{ OR } A \text{ SIMULTANEOUS } B$. However, since $A \text{ SIMULTANEOUS } B$ results in a zero because both events cannot occur at the same time, $A < B = A \text{ BEFORE } B$. This means that $(A \text{ PAND } B).(B \text{ PAND } A)$ is not a contradiction. However, according to Walker (2009), quantitatively analysing $(A \text{ PAND } B).(B \text{ PAND } A)$

$$Pr((A \text{ PAND } B).(B \text{ PAND } A)) \neq Pr(A \text{ PAND } B).Pr(B \text{ PAND } A)$$

$$Pr((A \text{ PAND } B).(B \text{ PAND } A)) = Pr(A \text{ SIMULTANEOUS } B)$$

The POR gate, as will be discussed later in this literature, is used to represent situations where an output event is triggered when its first input event occurs before the occurrence of a subsequent input event or where only the first input event alone occurs without the occurrence of the subsequent input event. Another limitation of the DFT technique, discussed in Edifor *et al.* (2012), is its inability to concisely model the Priority-OR scenario. Edifor *et al.* (2012) conclude that evaluating several POR scenarios with DFTs can be error-prone and cumbersome to model.

2.4 Temporal Fault Tree (TFT) Analysis

A more recent enhancement of classical FTA is temporal fault tree (TFT) analysis. TFT analysis discussed in this thesis refers to fault trees with temporal behaviours encoded in their structure. Palshikar (2002) provides some TFT analyses using novel temporal gates by which the sequential ordering of event occurrences of a fault tree can be described. These logical descriptions are then formalised using the Past-oriented Linear Propositional Temporal Logic (PLTLP). Palshikar's TFT is therefore a formal description of a fault tree with temporal behaviours. Though his technique provides qualitative analysis, it is useful as a diagnostic tool for analysing a system that has failed.

A more recent TFT technique is Pandora (Walker & Papadopoulos 2006; Walker & Papadopoulos 2007; Walker *et al.* 2007; Walker & Papadopoulos 2008; Walker 2009). The word ‘PANDORA’ was brewed from two words: ‘PAND’, for Priority-AND, and ‘ORA’, meaning ‘hour or time’ in Greek. Therefore ‘Pandora’ means ‘the time of PAND’. It is a technique recently developed by the University of Hull for performing qualitative analysis of temporal fault trees. Pandora was developed to provide solutions to some of DFT’s limitations while maintaining SFT’s simple approach to qualitative analysis. Pandora’s aim is to define clearer and more precise dynamic gates similar to that of DFT while maintaining the easy-to-use and easy-to-understand nature of classical FTA. For the remaining sections of this thesis, unless otherwise stated, the term TFT refers to Pandora and not Palshikar’s definition.

In addition to the Boolean gates (OR and AND) employed in SFT analysis, Pandora introduces two novel gates – Simultaneous-AND (SAND) and Priority-OR (POR). It also uses a modified PAND gate derived from Fussell *et al.* (1976). The AND and OR gates have been described in earlier sections; however, the descriptions of Pandora’s PAND, SAND and POR follows. Analogous to MCS in SFT is the MCSQ (Minimal Cut Sequence) in Pandora. A MCSQ is a smallest combination/sequence of basic events related by temporal/dynamic gates that causes the occurrence of a top-event. In other words, MCSQs are sequences of basic events necessary and sufficient to cause a top-event.

Priority-AND Gate

Pandora retains the basic definition of the original PAND gate: “failures [of input events] must occur in a specific sequence in order to generate the output event” (Fussell *et al.* 1976). Just like the traditional PAND gate, Pandora’s PAND gate is also true if and only if all its inputs are true and the input events have occurred in a particular sequence – usually from left-to-right with the rightmost occurring first. However, Pandora PAND gate differs slightly from the original PAND gate in DFTs. In Pandora (Walker 2009):

- the PAND gate is ‘exclusive’. Meaning, its inputs cannot occur at the same time but strictly in a particular sequence.
- temporal relations are mutually exclusive, so expressions like $(X \text{ PAND } Y) \text{ AND } (Y \text{ PAND } X)$ are impossible and will always be false.

Therefore the PAND is used to model the scenario where an event occurs strictly before another or other events; it is false for any simultaneous occurrence of input events.

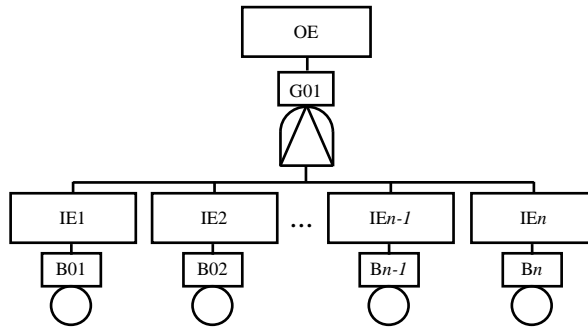


Figure 2.4-1: A PAND gate with n input events

Figure 2.4-1 is a graphical representation of a PAND gate with n input events $IE1, IE2 \dots IEn-1, IEn$ and an output event OE . OE will be triggered only when $IE1$ occurs strictly before $IE2$, $IE2$ occurs strictly before $IE3 \dots IEn-1$ occurs strictly before IE_n . The symbolic representation of PAND for both qualitative and quantitative analyses is ' $<$ '. Therefore $A < B$ is A PAND B which means A occurs strictly before B.

Simultaneous-AND Gate

The SAND gate is modelled to represent simultaneous occurrence of events. It does not overlap with the definition of the PAND gate in any way; they are mutually exclusive hence both cannot be true at the same time. Therefore $(X \text{ PAND } Y) \text{ AND } (X \text{ SAND } Y)$ will result in a FALSE value. Figure 2.4-2 is a diagrammatic representation of a SAND gate with n events. OE is an output event which is fired if $IE1, IE2, \dots, IEn-1, IEn$ are fire at exactly the same time. SAND is represented by the symbol '&'; $A \& B$ is A SAND B meaning, A and B occur at exactly the same time.

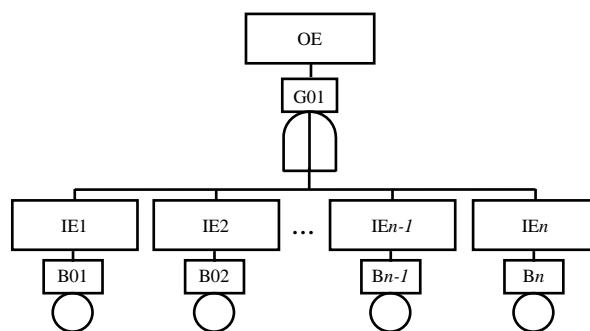


Figure 2.4-2: A SAND gate with n input events

Priority-OR Gate

Pandora introduces the POR gate to model situations where an output event occurs when the following two conditions are met:

- a) all the input events occur in a specific order. These events occur strictly; no two events occur at the same time.

- b) the first event alone occurs (subsequent events need not occur).

Therefore, for any two events, the POR output event occurs when the first input event occurs strictly before the second event or the first input event occurs and the subsequent event does not. POR is similar to a temporal version of an OR gate by being true when at least one of its input events is true. However, POR differs slightly because it attaches higher priority to its leftmost input. The POR gate is slightly similar to the PAND gate because priority is given to the first input (preceding the others) but it becomes true as soon as this first input becomes true; unlike the PAND gate that remains false until all its input events have occurred before it turns true. Classical examples of the use of the POR gate are presented in Walker and Papadopoulos (2007) and Edifor *et al.* (2012). Figure 2.4-3 is a fault tree with a POR gate that triggers *OE* when *IE1* all n events occur in the particular sequences they appear (left-to-right) or when *IE1* occurs and the remaining events have not yet occurred.

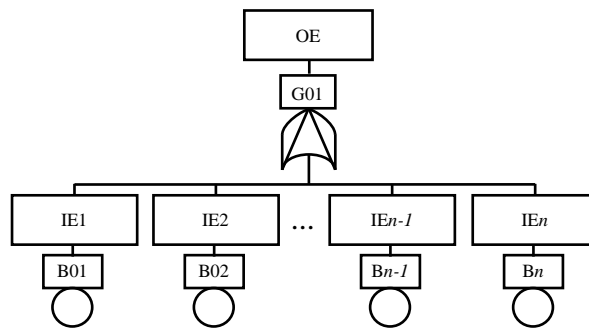


Figure 2.4-3: A POR gate with n input events

2.4.1 Qualitative Analysis

It is evident that standard Boolean gates and laws used in SFT analysis cannot adequately and sufficiently be used for TFT analysis because they make no provision for temporal ordering of events. As discussed earlier in the limitations of SFTs, any improvement to them must have the capability of considering temporal behaviours and must have laws for comprehensively analysing temporal gates logically.

Pandora presents solutions to this limitation by providing a formidable logical structure for presenting and analysing fault trees with temporal behaviours. It does this by the formulation of novel temporal laws. Pandora formulates some 143 novel laws (Walker 2009) in addition to the traditional Boolean laws to analyse its gates (both Boolean and temporal). These novel laws can be proved with the Temporal Truth Table (TTT). TTTs are analogous to the Boolean Truth Tables but contain values other than zero and one. These values are known as **sequence values** (Walker 2009) – they represent the sequen-

tial order with which events occur and not necessarily the exact time within which they occur. Some of the temporal laws are based on classical Boolean laws while others are completely novel. Details of the description of these laws are found in Walker’s PhD thesis (Walker 2009). Some of Pandora’s laws worth noting are the Completion Laws:

1st Completion Law: $A . B \Leftrightarrow A < B + A \ \& \ B + B < A$

2nd Completion Law: $A + B \Leftrightarrow A | B + A \ \& \ B + B | A$

3rd Completion Law: $A \Leftrightarrow B < A + A \ \& \ B + A | B$

Pandora has two primary distinct techniques for qualitative TFT analysis: Archimedes and Euripides. A third technique is the combination of both Archimedes and Euripides.

Euripides

Euripides (Walker 2009) is a law-based deductive technique that consists of four main stages. The first is the Binarboreal state, which is a recursive depth-first process that translates any gate of more than two inputs into “a nested series of the same type of gate” (Walker 2009), each with exactly two inputs. The second stage is the Flattening stage, where a repeated depth-first use of some classical Boolean laws is used to flatten (expand cut sequences breadthwise) the resulting tree from the Binarboreal stage into a structure called the Hierarchical Temporal Form (HTF) (Walker 2009). Flags are then set for each gate, to show that they have been flattened so that they are not re-flattened later. The gates are then rearranged and, based on the type of gate appropriate laws are applied to them to produce a binary HTF.

The third stage, Encapsulation stage or Doubletiser creates doublets (Walker *et al.* 2007), which are compositions of two basic event expressions and a temporal operator using temporal laws (Walker and Papadopoulos 2007; Walker 2009). The last stage, which is also the most relatively difficult, is the Minimisation stage. At this stage, cut sequences are minimised (reduced into their minimal forms) as far as possible by performing reductions within cut sequences and comparison of cut sequences.

Archimedes

Archimedes (Walker 2009) is an inductive methodical technique that converts fault trees into an alternative structure by enumerating all possibilities of cut sequences occurring and then generating possible sequences for a set of events. It commences by generating a dependency tree (logical tree designed to represent completion laws) from

a precedence tree (a representation of a branching timeline that shows all possible sequences for a set of basic events). Archimedes' evaluation of the dependency tree begins with assessing the expression for each basic temporal node, which represents every sequence of the events and every unique combination of possible sequence values. Actual sequence values are unnecessary in this process; however, their truth values are used to set a flag to true if the value is not zero and unset the flag to false when zero. The dependency tree is then traversed depth-first, counting children with their flags set. Parent nodes have their flags set if all their children have their flags set. The process is carefully done to eliminate duplicates. Upon completion of the evaluation, the dependency tree is then created using the completion laws to reduce the total number of MCSQs as much as possible.

Archimedes and Euripides

Archimedes and Euripides can be used to complement each other by harnessing their strengths for TFT analysis (Walker 2009). To achieve this, Euripides is used in obtaining the cut sequences (because of its efficacy at doing this) and these cut sequences are then analysed separately with Archimedes (because it is competent in appropriately analysing a smaller number of events).

2.4.2 TFT Quantitative Analysis

The importance of quantitative analyses has been discussed in section 2.2.2. They provide probabilities for the failure of systems, which helps determine the relative contribution of components or MCSQs and their contributions to the top-event occurrence. Pandora is a more recent technique, which provides only qualitative analysis of temporal fault trees. Therefore, unfortunately, unlike DFT, it does not provide any quantitative solution to its logical results. Techniques to quantify Pandora will be an improvement of Pandora and will have to satisfy the following conditions:

- Be able to provide quantitative analysis of each novel gate in Pandora
- Be capable of providing quantitative analysis of various combinations of both temporal and Boolean gates

2.4.3 Limitations of TFT

Pandora's major limitation is its inability to quantify TFTs. The solid qualitative framework it provides, in addition to other benefits, is an input for quantitative analysis. For a full analysis of Pandora, quantitative analysis must be possible. Techniques for quantifying Pandora may be formulated from Calculus, Markov Chains, Monte Carlo

simulation, Bayesian Nets and the likes. The main aim of this thesis is the exploration of previous fault tree techniques or formulation of new techniques that will support the quantitative analysis of Pandora.

Another area of improvement is the efficiency of its logical techniques: Euripides and Archimedes. Even though Euripides works efficiently through reduction and contradiction, it must be enhanced to detect completion-based reductions. Also it faces performance problems when the number of events increases because the minimisation stage uses an algorithm that depends on creating prime numbers for each basic event. If this prime number technique is not used in minimisation, the volume of checking required during minimization is greater: hitting harder at performance. Dependency trees depict all possible cut sequences. For increasing numbers of events, this tree grows exponentially, increasing computational costs. This is unfortunately the downside of using Archimedes.

2.5 Review of Quantitative Techniques

So far in this thesis, various techniques for quantitatively evaluating fault trees (static or dynamic) have been discussed. We group these into four main headings for comparative analysis: Monte Carlo simulations, Markov chain based solutions, Bayesian network based solutions and algebraic solutions. Table 2.5-1 contains a summary of the advantages and limitations of these techniques.

From Table 2.5-1, it is obvious that one of the more scalable techniques is Monte Carlo simulation. Unlike Markov chains-based techniques which suffer performance restrictions (due to state-space explosion) when analysing intricate systems, Monte Carlo simulations can be used to evaluate such systems – repairable or non-repairable – of various network configurations and featuring various failure distributions. The unfortunate drawback of the simulation technique is its ‘need’ to run several trials to enable it to estimate more accurate results. Though recent research efforts have been made to improve the performances of Markov chain-based and Monte Carlo techniques, it is still clear that the latter is far more scalable than the former.

Bayesian networks are efficient in representing dynamic behaviours inherent in modern systems. They can also be used for modelling complex safety-critical systems with different network configurations. However, unlike Monte Carlo or the algebraic techniques, they are incapable of analysing systems featuring different failure distributions.

Table 2.5-1: Comparison of Quantitative Techniques

Technique	Advantages	Limitations
Monte Carlo (Wang and Pham 1997)	<ol style="list-style-type: none"> 1. Not restricted to any system distribution. 2. Can be applied to most network configurations 3. Appropriate for modelling and evaluating complex systems. 4. Can incorporate priori information into its simulation using Bayes method. 5. Can evaluate repairable and non-repairable systems. 	<ol style="list-style-type: none"> 1. May require high computing resources for more accurate results. 2. Significant digit number of confidence limits is small.
Markov (Vesely et al. 2002; Merle 2010)	<ol style="list-style-type: none"> 1. Can evaluate repairable and non-repairable systems. 2. Appropriate for relatively small systems. 3. Provides exact results (unlike estimations produced by simulations). 	<ol style="list-style-type: none"> 1. Restricted to exponential distribution. 2. Susceptible to state-space explosion. 3. Error-prone and cumbersome to design for large fault trees.
Bayesian (Torres-Toledano and Sucar 1998; Weber and Jouffe 2003; Merle 2010)	<ol style="list-style-type: none"> 1. Powerful technique for explicitly representing dependencies. 2. Can be applied to most network configurations (series, parallel, bridge etc.) 3. Can be used in modelling complex systems. 	<ol style="list-style-type: none"> 1. Restricted to Gaussian distribution with truncated exponentials. 2. Restricted to DFT and SFT evaluations
Algebraic (Merle 2010)	<ol style="list-style-type: none"> 1. Not restricted to any system distribution. 2. Can be used to evaluate relatively larger systems. 3. Provides exact results (unlike estimations produced by simulations). 	<ol style="list-style-type: none"> 1. Restricted to non-repairable events. 2. Restricted to DFT and SFT evaluations

The algebraic technique, just as Monte Carlo simulation, can be used to analyse complex systems featuring different failure distributions. However, though the former provides exact results, the latter provides approximations. Monte Carlo simulation has its advantages over the algebraic technique too: it can be used to model and evaluate non-repairable systems and is not restricted to any network configuration.

From the above comparative analysis, it is obvious that the most scalable techniques are Monte Carlo simulation and the algebraic technique. For this reason, these two techniques will be considered, in the remaining chapters of this thesis, for the quantitative analysis of Pandora. It must be noted that the algebraic technique provides formulae for evaluating DFTs quantitatively. These formulae have been produced from calculus (first-principle). They provide quantitative solutions for the PAND gate and a POR gate with two events. This thesis explores the quantitative analysis of Pandora's gates using Calculus and Monte Carlo simulation.

Chapter Three

TEMPORAL QUANTITATIVE ANALYSIS

This chapter contains the main contributions of this thesis. Firstly, it describes the behavioural, timing and analytical frameworks that make the quantitative analysis of the SAND, pSAND, ePAND and POR gates possible. Secondly, it provides generic Monte Carlo simulation models for these gates. Thirdly, it prescribes a precedence order for evaluating MCSQs and describes methods for evaluating MCSQs containing combinations of two or more gates. Finally, it provides Monte Carlo algorithms for modelling, simulating and evaluating combinations of component failures and total system failures with various component failure distributions. It must be noted that apart from Monte Carlo simulation, all analytical solutions are restricted to exponential distribution of component failures.

In this chapter, various formal semantic expressions of Pandora are used. Details of these expressions and what they mean are provided in Walker et al., (2007). Some of the symbols (used in the semantic expressions) worth noting are:

$o(E)$ a set of all possible event orderings for a given set of events E .

$Po(E)$ power set of an event ordering o .

$pre(o)$ a set of event orderings preceding event ordering o in a precedence tree.

3.1 SAND Gate Quantification

The SAND output event occurs when all its input events occurs at exactly the same time. All input events must occur and, without any form of priority, they must occur at exactly the same time.

3.1.1 Behavioural and Timing Models

The formal semantic temporal definition (Walker et al., 2007; Merle, 2010) of the SAND gate with all input events occurring at exactly the same time is:

$$\&: Po(E) \times Po(E) \rightarrow Po(E) \quad (3.1-1)$$

$$\forall X_1, X_2: Po(E) \bullet o \in X_1 \& X_2 \Leftrightarrow o \in X_1 \wedge o \in X_2 \wedge \forall r: pre(o) \bullet r \in X_1 \Leftrightarrow r \in X_2$$

Where,

$$t(X_1) < t(X_2) \Rightarrow t(X_1 \& X_2) = \emptyset \quad (3.1-2)$$

$$t(X_1) = t(X_2) \Rightarrow t(X_1 \& X_2) = t(X_2) \quad (3.1-3)$$

$$t(X_1) > t(X_2) \Rightarrow t(X_1 \& X_2) = \emptyset \quad (3.1-4)$$

The symbols ‘<’ and ‘>’ are the less than and greater than operators respectively. Fig. 3.1-1A, Fig. 3.1-1B and Fig. 3.1-1C below are the timing behaviours for (3.1-2), (3.1-3), (3.1-4) respectively.

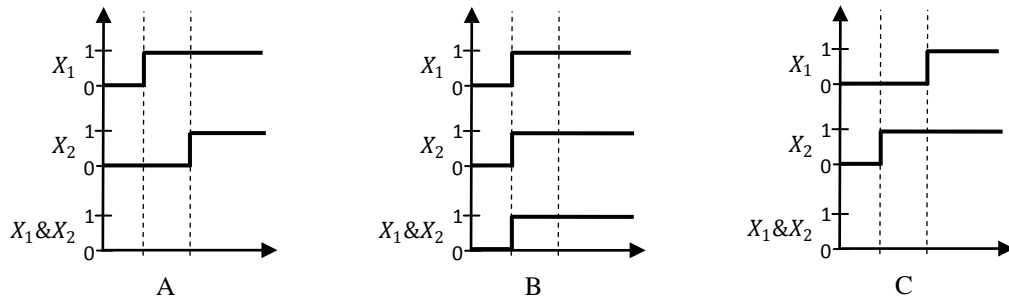


Figure 3.1-1: Timing models of an SAND gate

In Fig. 3.1-1A X_1 occurs before X_2 , $X_1 \& X_2$ does not occur. In Fig. 3.1-1B X_1 occurs at exactly the same time as X_2 , $X_1 \& X_2$ occur at $t(X_2)$ or $t(X_1)$. In Fig. 3.1-1C X_2 occurs before X_1 , $X_1 \& X_2$ does not occur. For the SAND gate, also known as the non-parameterized SAND, all input events occur at exactly the same time.

3.1.2 Analytical Model

In this section, we evaluate the SAND gate probability. We do so by evaluating $P(X_1 \& X_2)\{t\}$, where X_1 and X_2 occur exactly at the same time, t , from four expressions – a logical description and Pandora’s three Temporal Completion Laws. We then provide the SAND probability for n events.

From Logical Definition

Logically, the probability of two events occurring at exactly the same time is equal to the probability that both events occur and none occurs before the other. Thus,

$$\begin{aligned} X_1 \& X_2 &= X_1 \cdot X_2 \cdot (\text{NOT}(X_1 < X_2)) \cdot (\text{NOT}(X_2 < X_1)) \\ P(X_1 \& X_2)\{t\} &= P\left(X_1 \cdot X_2 \cdot (1 - (X_1 < X_2)) \cdot (1 - (X_2 < X_1))\right)\{t\} \\ &= P(X_1 \cdot X_2 - X_1 \cdot X_2 \cdot (X_2 < X_1) - X_1 \cdot X_2 \cdot (X_1 < X_2) \\ &\quad + X_1 \cdot X_2 \cdot (X_1 < X_2) \cdot (X_2 < X_1))\{t\} \end{aligned}$$

Using the Principle of Inclusion-Exclusion (2.2-22),

$$\begin{aligned}
P(X_1 \& X_2)\{t\} &= P(X_1 \cdot X_2 - X_1 \cdot X_2 \cdot (X_2 < X_1) - X_1 \cdot X_2 \cdot (X_1 < X_2))\{t\} \\
&+ P(X_1 \cdot X_2 \cdot (X_1 < X_2) \cdot (X_2 < X_1))\{t\} \\
&- P(X_1 \cdot X_2 - X_1 \cdot X_2 \cdot (X_2 < X_1) \\
&- X_1 \cdot X_2 \cdot (X_1 < X_2))\{t\} \cdot P(X_1 \cdot X_2 \cdot (X_1 < X_2) \cdot (X_2 < X_1))\{t\}
\end{aligned}$$

However,

$$P(X_1 \cdot X_2 \cdot (X_1 < X_2) \cdot (X_2 < X_1))\{t\} = \emptyset$$

because $X_1, X_2, X_1 < X_2$ and $X_1 < X_2$ cannot all occur – it is a logical contradiction. Also, from Pandora's absorption laws,

$$X_1 \cdot X_2 \cdot (X_1 < X_2) = (X_1 < X_2) \Rightarrow P(X_1 \cdot X_2 \cdot (X_1 < X_2))\{t\} = P(X_1 < X_2)\{t\}$$

Thus,

$$\begin{aligned}
P(X_1 \& X_2)\{t\} &= P(X_1)\{t\} \cdot P(X_2)\{t\} \\
&- [P(X_1 < X_2)\{t\} + P(X_2 < X_1)\{t\}]
\end{aligned} \tag{3.1-5}$$

From Completion Law 1

According to Pandora's Completion Law 1,

$$X_1 \cdot X_2 = (X_1 < X_2) + X_1 \& X_2 + (X_2 < X_1) \tag{3.1-6}$$

Thus,

$$P(X_1 \cdot X_2)\{t\} = P((X_1 < X_2) + X_1 \& X_2 + (X_2 < X_1))\{t\}$$

Using the (2.2-22),

$$\begin{aligned}
P(X_1 \cdot X_2)\{t\} &= P(X_1 < X_2)\{t\} + P(X_1 \& X_2)\{t\} + P(X_2 < X_1)\{t\} \\
&- P(X_1 < X_2)\{t\} \cdot P(X_1 \& X_2)\{t\} \\
&- P(X_1 < X_2)\{t\} \cdot P(X_2 < X_1)\{t\} \\
&- P(X_1 \& X_2)\{t\} \cdot P(X_2 < X_1)\{t\} \\
&+ P(X_1 < X_2)\{t\} \cdot P(X_1 \& X_2)\{t\} \cdot P(X_2 < X_1)\{t\}
\end{aligned}$$

However, the following terms result in logical contradictions; hence evaluate to zeroes.

$$\begin{aligned}
P(X_1 < X_2)\{t\}.P(X_1\&X_2)\{t\} &= \emptyset \\
P(X_1 < X_2)\{t\}.P(X_2 < X_1)\{t\} &= \emptyset \\
P(X_1\&X_2)\{t\}.P(X_2 < X_1)\{t\} &= \emptyset \\
P(X_1 < X_2)\{t\}.P(X_1\&X_2)\{t\}.P(X_2 < X_1)\{t\} &= \emptyset
\end{aligned}$$

Therefore,

$$\begin{aligned}
P(X_1\&X_2)\{t\} &= P(X_1)\{t\}.P(X_2)\{t\} \\
&\quad - [P(X_1 < X_2)\{t\} + P(X_2 < X_1)\{t\}]
\end{aligned} \tag{3.1-7}$$

From Completion Law 2

From Pandora's Completion Law 2,

$$X_1 + X_2 = X_1|X_2 + X_1\&X_2 + X_2|X_1$$

Therefore,

$$\begin{aligned}
P(X_1)\{t\} + P(X_2)\{t\} - P(X_1)\{t\}.P(X_2)\{t\} &= \\
P(X_1|X_2)\{t\} + P(X_1\&X_2)\{t\} + P(X_2|X_1)\{t\} - \\
P(X_1|X_2)\{t\}.P(X_1\&X_2)\{t\} - P(X_1|X_2)\{t\}.P(X_2|X_1)\{t\} - \\
P(X_1\&X_2)\{t\}.P(X_2|X_1)\{t\} + P(X_1|X_2)\{t\}.P(X_1\&X_2)\{t\}.P(X_2|X_1)\{t\}
\end{aligned}$$

For the same reasons as stated earlier, the remaining terms result in logical contradictions.

$$\begin{aligned}
P(X_1|X_2)\{t\}.P(X_1\&X_2)\{t\} &= \emptyset \\
P(X_1|X_2)\{t\}.P(X_2|X_1)\{t\} &= \emptyset \\
P(X_1\&X_2)\{t\}.P(X_2|X_1)\{t\} &= \emptyset \\
P(X_1|X_2)\{t\}.P(X_1\&X_2)\{t\}.P(X_2|X_1)\{t\} &= \emptyset
\end{aligned}$$

Therefore,

$$\begin{aligned}
P(X_1)\{t\} + P(X_2)\{t\} - P(X_1)\{t\}.P(X_2)\{t\} &= \\
P(X_1|X_2)\{t\} + P(X_1\&X_2)\{t\} + P(X_2|X_1)\{t\} -
\end{aligned}$$

Recalling the POR derivation,

$$\begin{aligned}
P(X_1)\{t\} + P(X_2)\{t\} - P(X_1)\{t\}.P(X_2)\{t\} &= \\
P(X_1 < X_2)\{t\} + P(X_1)\{t\}.[1 - P(X_2)\{t\}] + P(X_1\&X_2)\{t\} + \\
P(X_2 < X_1)\{t\} + P(X_2)\{t\}.[1 - P(X_1)\{t\}]
\end{aligned}$$

and hence,

$$\begin{aligned}
P(X_1 \& X_2)\{t\} &= P(X_1)\{t\} \cdot P(X_2)\{t\} \\
&\quad - [P(X_1 < X_2)\{t\} + P(X_2 < X_1)\{t\}]
\end{aligned} \tag{3.1-8}$$

From Completion Law 3

The Completion Law 3 in Pandora states that,

$$\begin{aligned}
X_1 &= X_2 < X_1 + X_1 \& X_2 + X_1 | X_2 \\
P(X_1)\{t\} &= P((X_2 < X_1) + (X_1 \& X_2) + (X_1 | X_2))\{t\}
\end{aligned} \tag{3.1-9}$$

Expanding the (3.1-9) using (2.2-22) and removing all contradictory terms produces,

$$\begin{aligned}
P(X_1)\{t\} &= P(X_2 < X_1)\{t\} + P(X_1 \& X_2)\{t\} + P(X_1 | X_2)\{t\} \\
&= P(X_2 < X_1)\{t\} + P(X_1 \& X_2)\{t\} + P(X_1 < X_2)\{t\} + \\
&\quad P(X_1)\{t\} - P(X_1)\{t\} \cdot P(X_2)\{t\}
\end{aligned}$$

Finally,

$$\begin{aligned}
P(X_1 \& X_2)\{t\} &= P(X_1)\{t\} \cdot P(X_2)\{t\} \\
&\quad - [P(X_1 < X_2)\{t\} + P(X_2 < X_1)\{t\}]
\end{aligned} \tag{3.1-10}$$

It is evident that (3.1-5), (3.1-7), (3.1-8), (3.1-10) are exactly the same and SAND is commutative, meaning,

$$P(X_1 \& X_2)\{t\} = P(X_2 \& X_1)\{t\}$$

Using (3.2-5), (2.1-13) and (2.1-14), the mathematical expression for a SAND gate with two events obtained from (3.1-5), (3.1-7), (3.1-8), (3.1-10) is,

$$\begin{aligned}
P(X_1 \& X_2)\{t\} &= F(X_1)\{t\} \cdot F(X_2)\{t\} \\
&\quad - \left\{ \int_0^t \{f(X_2)\{t\} \cdot F(X_1)\{t\}\} dy + \int_0^t \{f(X_1)\{t\} \cdot F(X_2)\{t\}\} dy \right\} \\
&= (1 - e^{(-\lambda_1)t}) \cdot (1 - e^{(-\lambda_2)t}) \\
&\quad - \left\{ \int_0^t \{X_2 e^{(-\lambda_2)y} \cdot (1 - e^{(-\lambda_1)y})\} dy \right. \\
&\quad \left. + \int_0^t \{X_1 e^{(-\lambda_1)y} \cdot (1 - e^{(-\lambda_2)y})\} dy \right\} \\
&= 0
\end{aligned}$$

It follows that,

$$P(X_1 \& X_2 \& \dots \& X_{n-1} \& X_n)\{t\} = 0$$

This result is consistent with the fact that the probability of two or more independent stochastic events occurring exactly the same time is zero (Merle, Roussel, Lesage & Bobbio 2010). The SAND gate may not have much significance in quantitative analysis of independent events, but as earlier discussed, it is beneficial in qualitative analysis. The focus of this thesis is the quantitative analysis of safety-critical systems using Pandora therefore, unless stated otherwise, all MCSQs containing the SAND gate would be assumed to evaluate to zero.

3.2 Non-Inclusive PAND Gate Quantification

The PAND output event occurs when all its input events occur in a particular sequence; the first input event occurs strictly before the second and the second occurs strictly before the third and so on. As earlier mentioned, the output of an exclusive PAND gate occurs if and only if its leftmost input event occurs strictly before subsequent input events – they do not occur at the same time. As a reminder, it must be noted that the inclusive-PAND (*i*PAND) and exclusive-PAND (*e*PAND) are logically different; they have different interpretations. The former occurs when its input events occur strictly one after another or at the same time while the later occurs when its input events occur strictly one after another. PAND from this point onwards refers to *e*PAND.

3.2.1 Behavioural and Timing Models

The formal semantic temporal definition (Walker et al., 2007; Merle, 2010) of the PAND gate is given as:

$$\prec: \mathbf{Po}(E) \times \mathbf{Po}(E) \rightarrow \mathbf{Po}(E) \quad (3.2-1)$$

$$\forall X_1, X_2: \mathbf{Po}(E) \bullet o \in X_1 < X_2 \Leftrightarrow o \in X_1 \wedge o \in X_2 \wedge \exists r: pre(o) \bullet r \in X_1 \wedge r \notin X_2$$

Where,

$$t(X_1) < t(X_2) \Rightarrow t(X_1 < X_2) = t(X_2) \quad (3.2-2)$$

$$t(X_1) = t(X_2) \Rightarrow t(X_1 < X_2) = \emptyset \quad (3.2-3)$$

$$t(X_1) > t(X_2) \Rightarrow t(X_1 < X_2) = \emptyset \quad (3.2-4)$$

Fig. 3.2-1A, Fig. 3.2-1B and Fig. 3.2-1C below are the timing behaviours for (3.2-2), (3.2-3) and (3.2-4) respectively.

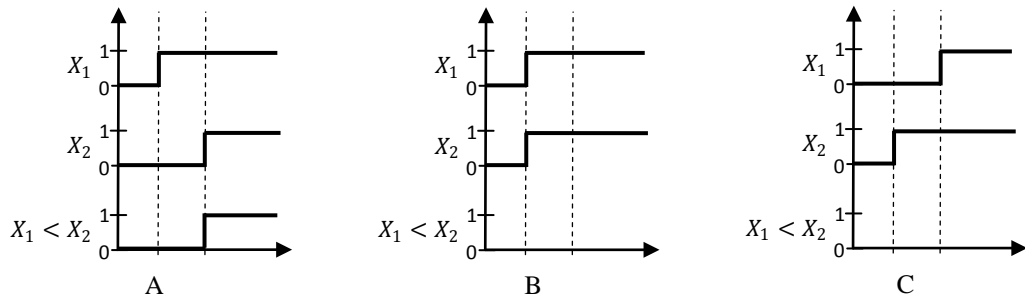


Figure 3.2-1: Timing models of an ePAND gate

It must be noted that the symbol ‘<’ used in the leftmost hand side of (3.2-2) is the less than operator and not the PAND operator. In Fig. 3.2-1A X_1 occurs before X_2 , $X_1 < X_2$ occurs at $t(X_2)$. In Fig. 3.2-1B X_1 occurs at the same time as X_2 , $X_1 < X_2$ does not occur. Finally, in Fig. 3.2-1C X_2 occurs before X_1 , $X_1 < X_2$ does not occur. The PAND gate output event is triggered when its entire input events occur and they do so in a particular sequence, one occurring after another.

3.2.2 Analytical Model

Fig. 3.2-2 is a graph of events $X_1, X_2, \dots, X_{n-1}, X_n$ and a time t . The ePAND probability, the probability that X_1 occurs strictly before X_2 and X_2 occurs strictly before X_3 and continuing in the same fashion until X_n , can be given as,

$$ePAND(X_1, X_2, \dots, X_{n-1}, X_n)\{t\} = iPAND(X_1, X_2, \dots, X_{n-1}, X_n)\{t\} - SAND(X_1, X_2, \dots, X_{n-1}, X_n)\{t\}$$

Meaning, the probability that X_1 occurs before X_2 is equal to the probability that X_1 occurs strictly before X_2 or the probability that both X_1 and X_2 occur at exactly the same time.

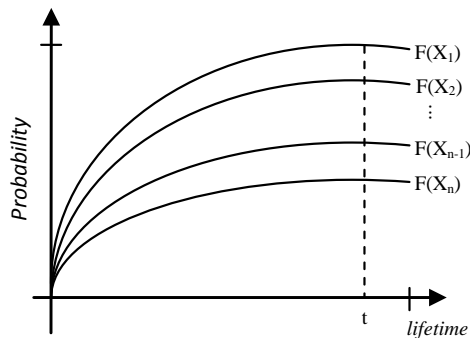


Figure 3.2-2: Graph of n events occurring at t

However, since the SAND probability of any n events is zero,

$$ePAND(X_1, X_2, \dots, X_{n-1}, X_n)\{t\} = iPAND(X_1, X_2, \dots, X_{n-1}, X_n)\{t\}$$

Therefore quantitatively, $ePAND$ is equal to $iPAND$; they both evaluate to zero. The significant difference between $iPAND$ and $ePAND$ lies in qualitative analysis where they have completely different meanings as explained in the previous section.

There are several techniques to evaluate the inclusive PAND gate. These including derivations from calculus (Fussell *et al.* 1976; Yuge and Yanagi 2008; Merle, Roussel, Lesage and Bobbio 2010), Markov Chains (Dugan *et al.* 1992), Stochastic Petri Nets (Marsan 1989), Bayesian Networks (Bobbio *et al.* 2001), Monte Carlo simulation (Durga Rao *et al.* 2009), and Poisson Stochastic Process (Eid 2011). In this thesis, all PAND gates, except otherwise stated, are quantitatively evaluated using Fussell *et al.*'s (1967) formula which states that for any N number of events,

$$P(N, N - 1, \dots, 2, 1)\{t\} = \prod_{i=1}^N \lambda_i \sum_{k=0}^N \left[\frac{e^{(a_k t)}}{\prod_{\substack{j=0 \\ j \neq k}}^N (a_k - a_j)} \right] \quad (3.2-5)$$

Where $a_0 = 0$ and $a_m = -\sum_{j=1}^m \lambda_j$ for $m > 0$.

3.2.3 Monte Carlo Solution

The algorithm below demonstrates how the PAND probability of n events can be estimated. Random numbers, R , and next Time-to-Failures, $TTFs$, are generated for all n events. If for all n random numbers R_x and R_{x+1} are less or equal to the actual probabilities of the event they represents and TTF_x is less than TTF_{x+1} , then a counter, S , is incremented. An estimation of the PAND probability is calculated by dividing S by T .

Algorithm 3.2-1: A Generic Monte Carlo simulation for the PAND gate

Require: $X[n]$

$S \leftarrow 0$

for $k = 1$ to T **do**

$F \leftarrow \text{true}$

$R[0] \leftarrow \text{NextRandomNumber}$

$TTF[0] = (1 / Pr(X[0])\{t\}) \times \mathbf{Log}(1 / (1 - R[0]))$

for $i = 1$ to n **do**

$R[i] \leftarrow \text{NextRandomNumber}$

$TTF[i] = (1 / Pr(X[0])\{t\}) \times \mathbf{Log}(1 / (1 - R[0]))$

```

if ( $R[i - 1] > Pr(X[i - 1])\{t\} \parallel R[i] > Pr(X[i])\{t\} \parallel TTF[i - 1] \geq TTF[i]$ ) then
     $F \leftarrow \text{false}$ 
end if
end for
if ( $F$ ) then
     $S \leftarrow S + 1$ 
end if
end for
return  $S/T$ 

```

3.3 POR Gate Quantification

The POR output event occurs when its first input event occurs before any subsequent input events occur or when the first input event occurs but the subsequent input events do not occur. The occurrence of the subsequent events does not matter; however, the priority does: the first event must occur strictly before all subsequent events.

3.3.1 Behavioural and Timing Models

The formal semantic temporal definition (Walker *et al.*, 2007; Merle, 2010) of the POR gate is:

$$| : \mathbf{Po}(E) \times \mathbf{Po}(E) \rightarrow \mathbf{Po}(E) \quad (3.3-1)$$

$$\forall X_1, X_2: \mathbf{Po}(E) \bullet o \in X_1 | X_2 \Leftrightarrow o \in X_1 \wedge \exists r: pre(o) \cup \{o\} \bullet r \in X_1 \wedge r \notin X_2$$

Where,

$$t(X_1) < t(X_2) \Rightarrow t(X_1 | X_2) = t(X_2) \quad (3.3-2)$$

$$t(X_1) = t(X_2) \Rightarrow t(X_1 | X_2) = \emptyset \quad (3.3-3)$$

$$t(X_1) > t(X_2) \Rightarrow t(X_1 | X_2) = \emptyset \quad (3.3-4)$$

$$t(X_1.NOT(X_2)) \Rightarrow t(X_1 | X_2) = t(X_1) \quad (3.3-5)$$

$$t(X_2.NOT(X_1)) \Rightarrow t(X_1 | X_2) = \emptyset \quad (3.3-6)$$

Again, it must be noted that the symbol ‘<’ used in the leftmost hand side of (3.3-2) is the less than operator and not the PAND operator. Fig. 3.3-1A, B, C, D and E below are the timing behaviours corresponding with (3.3-2), (3.3-3), (3.3-4), (3.3-5) and (3.3-6) respectively.

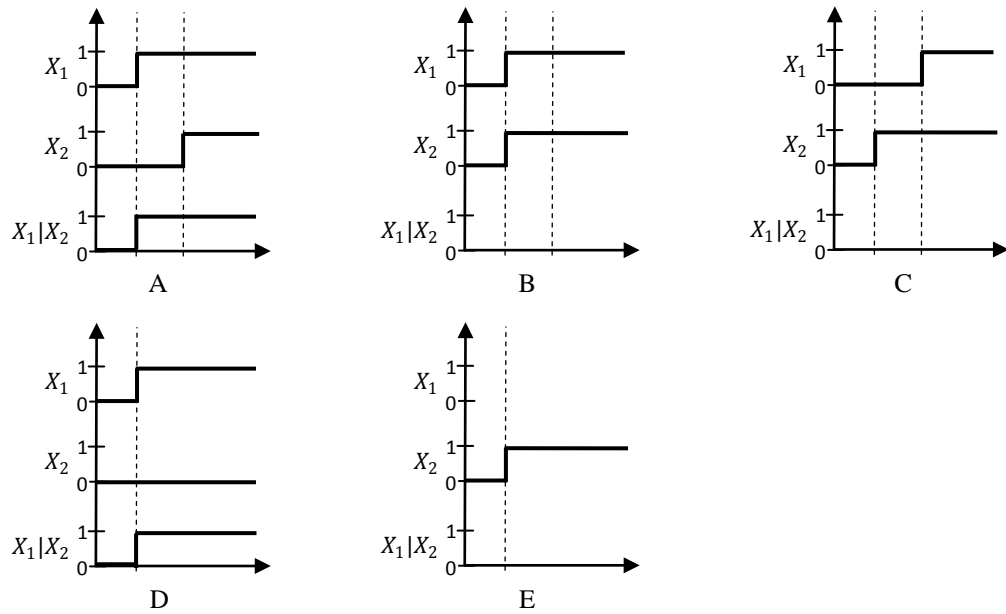


Figure 3.3-1: Timing models of a POR gate

In Fig. 3.3-1A X_1 occurs before X_2 , $X_1 | X_2$ occurs at $t(X_1)$. In Fig. 3.3-1B X_1 occurs at the same time as X_2 , $X_1 | X_2$ does not occur. Fig. 3.3-1C X_2 occurs before X_1 , $X_1 | X_2$ does not occur. In Fig. 3.3-1D X_1 occurs but X_2 does not, $X_1 | X_2$ occurs at $t(X_1)$. In Fig. 3.3-1E X_2 occurs but X_1 does not, $X_1 | X_2$ does not occur at all.

The POR gate occurs when an input event occurs and the subsequent event has not yet occurred.

3.3.2 Analytical Model

A POR gate can be thought of as being equivalent to $(X_1 \text{ PAND } X_2) \text{ OR } (X_1 \text{ AND NOT } (X_2))$. Meaning $X_1 / X_2 \Leftrightarrow X_1 < X_2 + X_1 \bullet \neg X_2$. The POR gate ensures that the fault tree remains coherent, unlike use of the NOT gate (Andrews 2000). In this thesis, it is assumed that any system under consideration is coherent, i.e., it cannot improve as a whole when one or more of its components fail (Esary & Proschan 1963).

Firstly, the analytical solution for the POR gate with only two events is derived from Markov Chains and Pandora's logical analysis. A generic evaluation of POR gates for multiple events is then deduced afterwards using calculus and some pre-existing logical laws (temporal encapsulation, binary and temporal associative laws) in Pandora.

Merle's Algebraic Solution

Merle (2010) provides an algebraic model for probabilistically analysing the PAND, Spare, and FDEP gates of a DFT. It equates the POR gate to a “Non-inclusive BEFORE”, represent by the symbol ‘◁’.

Using the definition of $f(X)\{t\}$ and $F(X)\{t\}$ in (2.1-13) and (2.1-14) respectively, Merle provides a probabilistic expression for the Non-inclusive BEFORE as:

$$\begin{aligned}
 P(X_1|X_2)\{t\} &= \int_0^t (f(X_1)\{t\}(1 - F(X_2)\{t\}))dy \\
 &= \int_0^t \left((\lambda_1 * e^{-\lambda_1*y}) \left(1 - (1 - e^{-\lambda_2*y}) \right) \right) dy \\
 &= \int_0^t (\lambda_1 * e^{-(\lambda_1+\lambda_2)y}) dy
 \end{aligned} \tag{3.3-7}$$

Derivation from Markov Chains

In reliability engineering, Markov models are widely used for analysing continuous time, discrete state scenarios (DoD 1998; Pukite & Pukite 1998). Fig. 3.3-2 represents a Markov model for the POR gate with two basic input events X_1 and X_2 which have failure rates λ_1 and λ_2 respectively. The arrowed lines represent the transition from one state to another and are labelled with the failure rate at which the transition occurred. The circles represent the states; failure states of the model (2 and 3) are shaded while non-failure states are not (1 and 4). Transition to state 4 is ignored because it does not lead to failure. With this in mind, at state 1, the system is fully functional and input event X_1 has not failed. At state 2, X_1 has failed but X_2 has not, nevertheless leading to total failure of the system. If X_2 subsequently fails (leading to state 3), the system remains in a failed state.

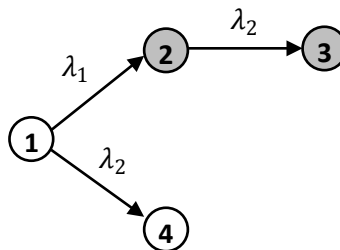


Figure 3.3-2: Markov model of a POR gate

The probability of being in state 1 at a particular time $t + dt$ is equal to that of being in state 1 at t and not transitioning to 2 during $(t, t + dt)$. Mathematically this can be expressed as:

$$\frac{\partial}{\partial t} P_1(t) = -(\lambda_1 + \lambda_2)P_1(t) \quad (3.3-8)$$

The probabilities of being in states 2, 3 and 4 are similarly given as:

$$\frac{\partial}{\partial t} P_2(t) = \lambda_1 P_1(t) - \lambda_2 P_2(t) \quad (3.3-9)$$

$$\frac{\partial}{\partial t} P_3(t) = \lambda_2 P_2(t) \quad (3.3-10)$$

$$\frac{\partial}{\partial t} P_4(t) = \lambda_2 P_1(t) \quad (3.3-11)$$

Solving (3.3-8), (3.3-9), (3.3-10), (3.3-11) gives:

$$P_1(t) = e^{-(\lambda_1 + \lambda_2)t}$$

$$P_2(t) = -\frac{\lambda_2}{\lambda_1 + \lambda_2} e^{-(\lambda_1 + \lambda_2)t} + \frac{\lambda_2}{\lambda_1 + \lambda_2}$$

$$P_3(t) = e^{-(\lambda_2)t} - e^{-(\lambda_1 + \lambda_2)t}$$

$$P_4(t) = \frac{\lambda_2}{\lambda_1 + \lambda_2} e^{-(\lambda_1 + \lambda_2)t} - e^{-(\lambda_2)t} + 1 - \frac{\lambda_2}{\lambda_1 + \lambda_2}$$

$$P(X_1|X_2)\{t\} = P_3(t) + P_4(t) = \frac{\lambda_1(1 - e^{-(\lambda_1 + \lambda_2)t})}{\lambda_1 + \lambda_2} \quad (3.3-12)$$

Derivation from Pandora's definition of POR

From the definition of a POR gate it is clear that the occurrence of a POR gate, e.g. X_1 / X_2 , is dependent on the occurrence of either of two cases, i.e., X_1 before X_2 (i.e., $X_1 < X_2$) and X_1 without X_2 (i.e., $X_1 \bullet \neg X_2$). Mathematically, $X_1 / X_2 = X_1 < X_2 + X_1 \bullet \text{NOT}(X_2)$. Thus by calculating the probabilities of these two cases, one can determine the probability of the POR gate as a whole by using the principle of inclusion-exclusion.

It must be noted that $\text{NOT}(X)$ is the probability that X does not occur and is equal to $1 - F(X)$.

$$P(X_1|X_2)\{t\} = P(X_1 < X_2)\{t\} + P(X_1 \bullet \text{NOT}(X_2))\{t\} - P((X_1 < X_2) * (X_1 \bullet \text{NOT}(X_2)))\{t\}$$

However, $P(X_1 < X_2 * X_1 \bullet NOT(X_2))\{t\}$ results in a logical contradiction because both terms cannot occur at the same time – X_2 cannot happen both after X_1 and not at all. Therefore $P(X_1 < X_2 * X_1 \bullet NOT(X_2))\{t\} = 0$, and thus:

$$\begin{aligned}
P(X_1|X_2)\{t\} &= P(X_1 < X_2)\{t\} + P(X_1 \bullet NOT(X_2))\{t\} \\
&= \int_0^t (f\{X_2\}(t) * F\{X_1\}(t)) dy \\
&\quad + F\{X_1\}(t) \cdot (1 - F\{X_2\}(t)) \\
&= \int_0^t ((\lambda_2 * e^{-\lambda_2*y}) * (1 - e^{-\lambda_1*y})) dy \\
&\quad + (1 - e^{-\lambda_1*t}) \cdot (1 - (1 - e^{-\lambda_2*t})) \\
&= \frac{\lambda_1(1 - e^{-(\lambda_1+\lambda_2)t})}{\lambda_1 + \lambda_2} \tag{3.3-13}
\end{aligned}$$

Deriving the Multiple POR Formula

Until now, in this thesis, the expression deduced for evaluating a POR gate is restricted to only two events. However, in the real world, there may be MCSQs that may contain more than one POR gate in succession. For this reason a generic analytical formula for such PORs gate is derived from first principles and some of Pandora’s laws.

Multiple POR Formula from First Principles

For any POR MCSQs with the expression $X_1 / X_2 | \dots | X_{n-1} / X_n$, and constant failure rates $\lambda_1, \lambda_2 \dots \lambda_{n-1}, \lambda_n$ respectively, the probability of this MCSQ is derived as:

$$\begin{aligned}
P(X_1|X_2 | \dots | X_{n-1}|X_n)\{t\} \\
&= \int_0^t (f(X_1)\{t\}(1 - F(X_2)\{t\}) \dots (1 - F(X_{n-1})\{t\})(1 \\
&\quad - F(X_n)\{t\})) dy \\
&= \int_0^t (\lambda_1(e^{-\lambda_1 y})(e^{-\lambda_2 y}) \dots (e^{-\lambda_{n-1} y})(e^{-\lambda_n y})) dy \\
&= \frac{\lambda_1 - \lambda_1(e^{-(\lambda_1+\lambda_2+\dots+\lambda_{n-1}+\lambda_n)t})}{(\lambda_1 + \lambda_2 + \dots + \lambda_{n-1} + \lambda_n)} \\
&= \frac{\lambda_1(1 - (e^{-(\sum_{i=1}^n \lambda_i)t}))}{\sum_{i=1}^n \lambda_i} \tag{3.3-14}
\end{aligned}$$

Multiple POR Formula from Induction Using Pandora's Laws

The multiple POR formula in (3.3-14) can also be derived from three of Pandora's laws:

1. Temporal Encapsulation Law
2. Binary Law and
3. Temporal Associative Law

We prove this by formulating logical expressions for MCSQs with two or more POR gates from these laws. From these logical expressions a generic probabilistic formula for the multiple POR evaluation is induced.

From Temporal Encapsulation Law

One of Pandora's Temporal Encapsulation Laws states that

$$X|Y|Z = X|Y.X|Z \quad (3.3-15)$$

Therefore,

$$X_1|X_2|X_3 = X_1|X_2.X_1|X_3$$

However, as discussed earlier;

$$X_1|X_2 = X_1 < X_2 + X_1 \cdot \neg X_2 \quad (3.3-16)$$

$$X_1|X_3 = X_1 < X_3 + X_1 \cdot \neg X_3 \quad (3.3-17)$$

Substituting (3.3-16) and (3.3-17) into (3.3-15) produces,

$$\begin{aligned} X_1|X_2|X_3 &= \{X_1 < X_2 + X_1 \cdot \neg X_2\} \cdot \{X_1 < X_3 + X_1 \cdot \neg X_3\} \\ &= X_1 < X_2 \cdot X_1 < X_3 + X_1 < X_2 \cdot X_1 \cdot \neg X_3 + X_1 \cdot \neg X_2 \cdot X_1 \\ &< X_3 + X_1 \cdot \neg X_2 \cdot X_1 \cdot \neg X_3 \end{aligned} \quad (3.3-18)$$

By applying the temporal absorption (3.3-19) and idempotent (3.3-20) laws to (3.3-18), (3.3-21) is obtained.

$$X \cdot (X < Y) = X < Y \quad (3.3-19)$$

$$X \cdot X = X \quad (3.3-20)$$

$$\begin{aligned} X_1|X_2|X_3 &= X_1 < X_2 \cdot X_1 < X_3 + X_1 < X_2 \cdot \neg X_3 + X_1 \\ &< X_3 \cdot \neg X_2 + X_1 \cdot \neg X_2 \cdot \neg X_3 \end{aligned} \quad (3.3-21)$$

However, when $X_1 < X_2 \cdot X_1 < X_3$ is fully minimised using Pandora's laws, it produces

$$X_1 < X_2 < X_3 + X_1 < X_3 < X_2 + X_1 < X_2 \& X_3$$

But, as already proven, technically, the SAND gate is statistically zero and therefore, $X_1 < X_2 \& X_3$ is ignored in the logical analysis. Thus:

$$\begin{aligned} X_1|X_2|X_3 &= X_1 < X_2 < X_3 + X_1 < X_3 < X_2 + X_1 < X_2 \cdot \neg X_3 + X_1 \\ &< X_3 \cdot \neg X_2 + X_1 \cdot \neg X_2 \cdot \neg X_3 \end{aligned} \quad (3.3-22)$$

From Binary Law

One of Pandora's Binary Laws states that:

$$X|Y|Z = X|(Y + Z) \quad (3.3-23)$$

Using (3.3-23) a logical expression similar to what is done for the Temporal Encapsulation Law can be derived. Substituting (3.3-23) into (3.3-16) produces,

$$X_1|X_2|X_3 = X_1 < (X_2 + X_3) + X_1 \cdot \neg(X_2 + X_3) \quad (3.3-24)$$

However, one of Pandora's Temporal Distributive Laws states that

$$X_1 < (X_2 + X_3) = X_1|X_2 \cdot X_1|X_3 \cdot (X_2 + X_3)$$

Therefore,

$$X_1|X_2|X_3 = X_1|X_2 \cdot X_1|X_3 \cdot (X_2 + X_3) + X_1 \cdot \neg(X_2 + X_3)$$

Employing (3.3-16) again,

$$\begin{aligned} X_1|X_2|X_3 &= X_1|X_2 \cdot X_1|X_3 \cdot (X_2 + X_3) + X_1 \cdot \neg(X_2 + X_3) \\ &= X_1 < X_2 \cdot X_1 < X_3 \cdot X_2 + X_1 < X_2 \cdot X_1 < X_3 + X_1 \\ &< X_2 \cdot X_1 \cdot X_2 \cdot \neg X_3 + X_1 < X_2 \cdot X_1 \cdot X_3 \cdot \neg X_3 + X_1 \\ &< X_3 \cdot X_1 \cdot X_2 \cdot \neg X_2 + X_1 \\ &< X_3 \cdot X_1 \cdot \neg X_2 + X_1 < X_2 \cdot X_1 \cdot X_3 \cdot \neg X_3 + X_1 \\ &< X_3 \cdot X_1 \cdot X_2 \cdot \neg X_2 + X_1 < X_3 \cdot X_1 \cdot \neg X_2 + \end{aligned} \quad (3.3-25)$$

Applying (3.3-19), (3.3-20) and Boolean Complementary Laws $X \cdot \neg X = 0$ to (3.3-25), (3.3-22) is produced.

From Temporal Associative Law

Finally, we show that the logical expression (3.3-22) can be derived from one of Pandora's Temporal Associative Law that states that,

$$X|Y|Z = (X|Y)|Z \quad (3.3-26)$$

Substituting (3.3-16) into (3.3-26) produces,

$$X_1|X_2|X_3 = (X_1 < X_2 + X_1 \cdot \neg X_2)|X_3 \quad (3.3-27)$$

Applying the Temporal Distributive Law in (3.3-28) to (3.3-27) produces (3.3-29)

$$(X + Y)|Z = (X|Z) + (Y|Z) \quad (3.3-28)$$

$$X_1|X_2|X_3 = (X_1 < X_2)|X_3 + (X_1 \cdot \neg X_2)|X_3 \quad (3.3-29)$$

However,

$$(X_1 < X_2)|x_3 = (X_1|X_2) \cdot (X_2|X_3) \quad (3.3-30)$$

Therefore,

$$\begin{aligned} (X_1 < X_2)|X_3 &= \{X_1 < X_2 + X_1 \cdot \neg X_2\} \cdot \{X_2 < X_3 + X_2 \cdot \neg X_3\} \\ &= X_1 < X_2 \cdot X_2 < X_3 + X_1 < X_2 \cdot \neg X_3 + X_1 \cdot \neg X_2 \cdot X_2 \\ &< X_3 + X_1 \cdot X_2 \cdot \neg X_2 \cdot \neg X_3 \end{aligned}$$

It must be noted that due to the Boolean Redundancy and Temporal Absorption Laws

$$X_1 \cdot \neg X_2 \cdot X_2 < X_3 = X_1 \cdot \neg X_2 \cdot X_2 \cdot X_2 < X_3 = 0$$

Hence,

$$(X_1 < X_2)|X_3 = X_1 < X_2 \cdot X_2 < X_3 + X_1 < X_2 \cdot \neg X_3 \quad (3.3-31)$$

Also,

$$\begin{aligned} (X_1 \cdot \neg X_2)|X_3 &= (X_1|X_3) \cdot (\neg X_2|X_3) \\ &= \{X_1 < X_3 + X_1 \cdot \neg X_3\} \cdot \{\neg X_2 < X_3 + \neg X_2 \cdot \neg X_3\} \\ &= X_1 < X_3 \cdot \neg X_2 < X_3 + X_1 < X_3 \cdot \neg X_2 \cdot \neg X_3 + \\ &\quad X_1 \cdot \neg X_3 \cdot \neg X_2 < X_3 + X_1 \cdot \neg X_2 \cdot \neg X_3 \\ &= X_1 < X_3 \cdot \neg X_2 < X_3 + X_1 \cdot \neg X_2 \cdot \neg X_3 \end{aligned} \quad (3.3-32)$$

Substituting (3.3-31) and (3.3-32) into (3.3-29) produces,

$$\begin{aligned} X_1|X_2|X_3 &= X_1 < X_2 \cdot X_1 < X_3 + X_1 < X_2 \cdot \neg X_3 + X_1 \\ &< X_3 \cdot \neg X_2 + X_1 \cdot \neg X_2 \cdot \neg X_3 \\ &= X_1 < X_2 < X_3 + X_1 < X_3 < X_2 + X_1 < X_2 \cdot \neg X_3 + X_1 \\ &< X_3 \cdot \neg X_2 + X_1 \cdot \neg X_2 \cdot \neg X_3 \end{aligned}$$

Proof by Induction

Now that the logical expression (3.3-22) has been derived from some of Pandora's laws, we derive the probabilistic evaluation of a multiple POR expression from mathematical induction. It has been previously proven that for n events, where n is 2,

$$P(X_1|X_2)\{t\} = P(X_1 < X_2 + X_1 \cdot \neg X_2)\{t\}$$

$$P(X_1|X_2)\{t\} = \frac{\lambda_1(1 - e^{-(\lambda_1+\lambda_2)t})}{\lambda_1 + \lambda_2}$$

and n is 3,

$$\begin{aligned} P(X_1|X_2|X_3)\{t\} \\ &= P(X_1 < X_2 < X_3 + X_1 < X_3 < X_2 + X_1 \\ &< X_2 \cdot \neg X_3 + X_1 < X_3 \cdot \neg X_2 + X_1 \cdot \neg X_2 \cdot \neg X_3)\{t\} \end{aligned}$$

$$P(X_1|X_2|X_3)\{t\} = \frac{\lambda_1(1 - e^{-(\lambda_1+\lambda_2+\lambda_3)t})}{\lambda_1 + \lambda_2 + \lambda_3}$$

Using both Boolean and Pandora's Laws, for $n = 4$,

$$\begin{aligned} P(X_1|X_2|X_3|X_4)\{t\} \\ &= P(X_1 < X_4 < X_3 < X_2 + X_1 < X_3 < X_4 < X_2 + X_1 \\ &< X_3 < X_2 < X_4 + X_1 < X_4 < X_2 < X_3 + X_1 < X_2 < X_4 \\ &< X_3 + X_1 < X_2 < X_3 < X_4 + X_1 < X_2 \cdot X_1 \\ &< X_3 \cdot \neg X_4 + X_1 < X_2 \cdot X_1 < X_4 \cdot \neg X_3 + X_1 < X_3 \cdot X_1 \\ &< X_4 \cdot \neg X_2 + X_1 < X_2 \cdot \neg X_3 \cdot \neg X_4 + X_1 \\ &< X_3 \cdot \neg X_2 \cdot \neg X_4 + X_1 \\ &< X_4 \cdot \neg X_2 \cdot \neg X_3 + X_1 \cdot \neg X_2 \cdot \neg X_3 \cdot \neg X_4)\{t\} \end{aligned}$$

$$P(X_1|X_2|X_3|X_4)\{t\} = \frac{\lambda_1(1 - e^{-(\lambda_1+\lambda_2+\lambda_3+\lambda_4)t})}{\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4}$$

Finally, any n number of events,

$$\begin{aligned}
& P(X_1|X_2| \dots |X_{n-1}|X_n)\{t\} \\
&= P \left(\sum_{i=2}^{(n-1)!} \left(C_{j=2}^n (X_1 < X_j) \right) \right. \\
&\quad \left. + \left[\sum_{i=2}^n \left(\prod_{\substack{j=2 \\ j \neq i}}^n X_1 < X_j \right) \cdot \neg X_i \right] + \dots \right. \\
&\quad \left. + \left[\sum_{i=2}^n X_1 < X_i \cdot \left(\prod_{\substack{j=2 \\ j \neq i}}^n \neg X_j \right) \right] + \left[X_1 \cdot \left(\prod_{j=2}^n \neg X_j \right) \right] \right) \{t\}
\end{aligned}$$

Where,

$$C_{j=2}^n (X_1 < X_j) = X_1 < X_2 \cdot X_1 < X_3 \cdot \dots \cdot X_1 < X_{n-1} \cdot X_1 < X_n$$

Therefore,

$$\begin{aligned}
& P(X_1|X_2| \dots |X_{n-1}|X_n)\{t\} \\
&= P \left(\sum_{i=2}^{(n-1)!} \left(C_{j=2}^n (X_1 < X_j) \right) \right. \\
&\quad \left. + \left[\sum_{i=2}^n \left(\prod_{\substack{j=2 \\ j \neq i}}^n X_1 < X_j \right) \cdot (1 - F\{X_i\}(t)) \right] + \dots \right. \\
&\quad \left. + \left[\sum_{i=2}^n X_1 < X_i \cdot \left(\prod_{\substack{j=2 \\ j \neq i}}^n (1 - F\{X_j\}(t)) \right) \right] \right. \\
&\quad \left. + \left[F\{X_1\}(t) \cdot \left(\prod_{j=2}^n (1 - F\{X_j\}(t)) \right) \right] \right) \{t\}
\end{aligned}$$

$$\begin{aligned}
& P(X_1|X_2| \dots |X_{n-1}|X_n)\{t\} \\
&= \frac{\lambda_1 \cdot (e^{-(\lambda_1 + \lambda_2 + \dots + \lambda_{n-1} + \lambda_n)t}) \cdot (-1 + e^{(\lambda_1 + \lambda_2 + \dots + \lambda_{n-1} + \lambda_n)t})}{(\lambda_1 + \lambda_2 + \dots + \lambda_{n-1} + \lambda_n)}
\end{aligned}$$

$$P(X_1|X_2| \dots |X_{n-1}|X_n)\{t\} = \frac{\lambda_1 \left(1 - (e^{-(\sum_{i=1}^n \lambda_i)t})\right)}{\sum_{i=1}^n \lambda_i} \quad (3.3-33)$$

The multiple POR expression derived from Pandora's laws (3.3-33) is exactly the same as that derived from First Principles (3.3-14). This validates the multiple POR formula and demonstrates the uniformity and correctness of Pandora's logical laws.

3.3.3 Monte Carlo Solution

To formulate a Monte Carlo solution for the POR gate, a model gate needs to be constructed. Upon careful examination of Pandora's laws, one of its Binary Laws (3.3-34) promises to be a good foundation for modelling the POR gate.

$$\begin{aligned} P(X_1|X_2| \dots |X_{n-1}|X_n)\{t\} \\ = P\left(X_1 \left| \left((X_2 + X_3) \dots + X_{n-1} \right) + X_n \right.\right)\{t\} \end{aligned} \quad (3.3-34)$$

The right hand side of the above expression can be divided into two parts in relation to the POR gate:

1. X_1 and
2. $((X_2+X_3) \dots +X_{n-1})+X_n$ which is basically a summation of all events from X_2 to X_n .

If X_1 is event A and $((X_2+X_3) \dots +X_{n-1})+X_n$ is B , the POR probability can be computed with the following model.

1. Generate two random numbers, $R1$, $R2$ to simulate events A and B respectively.
2. If $R1 \leq A$ and $R2 \leq B$ and TTF of A is less than TTF of B , or $R1$ is less or equal to A but $R2$ is greater than B then a pre-set counter is incremented.
3. The above steps are repeated a large number of times.
4. The POR probability is evaluated by dividing the counter by the number of times the steps were repeated.

Using the definition of $R1$, $R2$, A and B above, Algorithm 3.3-1 is a Monte Carlo model for a multiple POR gate where S is a counter, $X[0]$ is A , m is $P(B)\{t\}$, T is the number of trials and TTF1 and TTF2 are next times-to-failure of A and B respectively.

Algorithm 3.3-1: A Generic Monte Carlo simulation for the POR gate

Require: $X[n]$

```

 $S \leftarrow 0$ 
 $TTF1 \leftarrow 0, TTF2 \leftarrow 0$ 
 $R1 \leftarrow 0, R2 \leftarrow 0$ 
for  $k = 1$  to  $T$  do
     $m \leftarrow 1$ 
    for  $i = 1$  to  $n$  do
         $m \leftarrow 1 - Pr(X[i])\{t\}$ 
    end for
     $m \leftarrow 1 - m$ 
     $R1 \leftarrow NextRandomNumber$ 
     $R2 \leftarrow NextRandomNumber$ 
     $TTF1 \leftarrow (1 / Pr(X[0])\{t\}) \times \mathbf{Log}(1 / (1 - R1))$ 
     $TTF2 \leftarrow (1 / m) \times \mathbf{Log}(1 / (1 - R2))$ 
    if  $((R1 \leq F(X[0])\{t\} \ \&\& \ R2 \leq m \ \&\& \ TTF1 < TTF2) \ || \ (R1 \leq F(X[0])\{t\} \ \&\& \ R2 > m))$  then
         $S \leftarrow S + 1$ 
    end if
end for
return  $S/T$ 

```

3.4 Parameterised-SAND Gate Quantification

In the previous section, the SAND gate was discussed and it was proven that, quantitatively, it evaluates to zero. However, some events, known as nearly simultaneous events, will trigger the occurrence of an output event if they should happen within a relatively short period or fraction of time – i.e., within a given interval. In other words, the output event of a nearly simultaneous gate will occur if the first input event occurs and the subsequent event occurs within a subsequent interval of time. The output event should occur if the sequence of the events were vice versa. In this thesis, a novel gate is introduced to represent this nearly simultaneous scenario. This new gate is known as the parameterised-SAND or pSAND.

A typical example of a pSAND scenario is evident in the case of the front wheel braking system of a car. If the left front wheel brakes of the car inadvertently activate, the vehicle begins to veer left. Conversely, if right front wheel brakes commit inadvertently, the vehicle veers to the right. Any of these failures can be potentially dangerous because they can alter the dynamics of the car causing the vehicle to veer off the road into a bush, river, pedestrian etc. or causing the vehicle to enter the path of oncoming vehicles.

However, if both brakes fail within a fraction of a second, this may cause harsh braking, which, if should occur on a motorway, could be critical. After this fraction of time, if an accident (top-event) has not occurred, the driver of the vehicle could follow some safety procedures, or the vehicle, if capable could automatically do so after detecting the failure of brakes.

Another classical example is the gas, burner and water sprinkler scenario. Assume a large scientific oven has a burner, which provides ignitions to a gaseous fuel source for burning metals at high temperature. This combustion chamber also has a sprinkler system, which is automatically activated after a fraction of time to put out fire in case of inadvertent combustion.

It must be noted that this combustion system is different from the popular gas leak/combustion systems formalised in Chaochen *et al.* (1991), Gorski and Wardzinski (1996) and Hansen *et al.* (1998) in two main areas. Firstly, the top event of the gas/leak combustions is usually an explosion due to the gas leak concentration and ignition; however, the top-event of the pSAND scenarios is inadvertent combustion due to the mere presence of gas (not necessarily a concentration leading to explosion) and ignition.

Secondly, the pSAND gate is commutative, so no matter the sequence of the events, the top event will occur provided one occurs first and the second occurs within a specified time interval. This cannot be said of the traditional gas leak system because if the gas leaks to a concentration before the ignition, then there will be an explosion. However, if the ignition fails by commission before the gas leaks, there will not be an explosion (top-event will not occur); there will be mere continuous combustion.

An extreme use of the pSAND gate is its application in supply chains where the duration is not a fraction of seconds but rather hours. For example, a top event may occur if the deliveries of two goods are not made within a couple of hours, after which a mitigation procedure is followed.

In this thesis, novel symbols for pSAND are modelled for qualitative and quantitative analyses. In addition, the original SAND gate is slightly extended to accommodate the pSAND's requirements, but not redefined entirely.

Semantics of pSAND:

All input events of the pSAND gate must occur and they must do so within a relatively short interval of duration 'd', which starts with the first input event to occur. The pSAND is therefore false if any of its inputs do not occur or if they occur outside the interval, i.e., the time between the first and last input to occur is more than d.

Since pSAND is a light modification of the SAND gate, the graphical symbol of the SAND gate Fig. 3.4-1A is retained for the situation where $d=0$: input events occur at exactly the same time. Alternatively, Fig. 3.4-1B can also be used to represent the same scenario where $d=0$. Fig. 3.4-1C on the contrary represents a situation where the input events are nearly simultaneous with an interval or duration, d , between them and $d>0$.

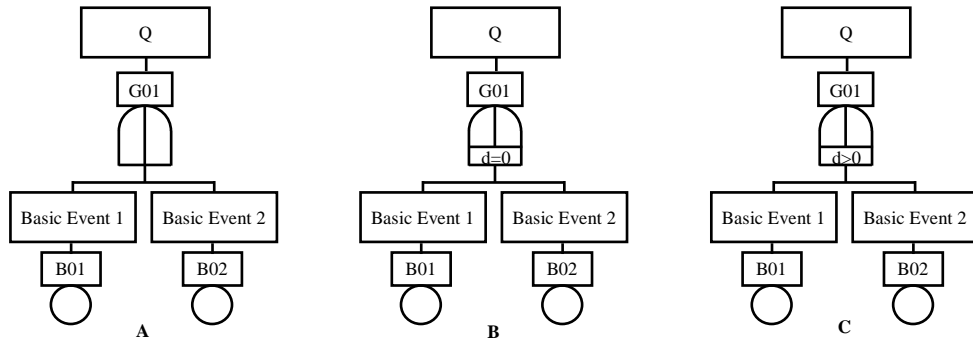


Figure 3.4-1: pSAND graphical representations

3.4.1 Behavioural and Timing Models

The formal semantic temporal definition of the pSAND gate given a specific duration, d , within which its input events occur is given as:

$$\&_d: \mathbf{Po}(E) \times \mathbf{Po}(E) \rightarrow \mathbf{Po}(E) \quad (3.4-1)$$

$$\forall X_1, X_2: \mathbf{Po}(E) \bullet o \in X_1 \&_d X_2 \Leftrightarrow o \in X_1 \wedge o \in X_2 \wedge \forall r: \text{pre}(o) \bullet r \in X_1 \Leftrightarrow r \in X_2$$

Where,

$$\{t(X_1) < t(X_2)\} \text{ AND } \{t(X_2) - t(X_1) \leq d\} \Rightarrow t(X_1 \&_d X_2) = t(X_2) \quad (3.4-2)$$

$$\{t(X_1) < t(X_2)\} \text{ AND } \{t(X_2) - t(X_1) > d\} \Rightarrow t(X_1 \&_d X_2) = \emptyset \quad (3.4-3)$$

$$\{t(X_1) > t(X_2)\} \text{ AND } \{t(X_1) - t(X_2) \leq d\} \Rightarrow t(X_1 \&_d X_2) = t(X_1) \quad (3.4-4)$$

$$\{t(X_1) > t(X_2)\} \text{ AND } \{t(X_1) - t(X_2) > d\} \Rightarrow t(X_1 \&_d X_2) = \emptyset \quad (3.4-5)$$

$$t(X_1) = t(X_2) \Rightarrow t(X_1 \&_d X_2) = t(X_2) \quad (3.4-6)$$

Fig. 3.4-2A, Fig. 3.4-2B, Fig. 3.4-2C, Fig. 3.4-2D and Fig. 3.4-2E below are the timing behaviours for (3.4-2), (3.4-3), (3.4-4), (3.4-5) and (3.4-6) respectively.

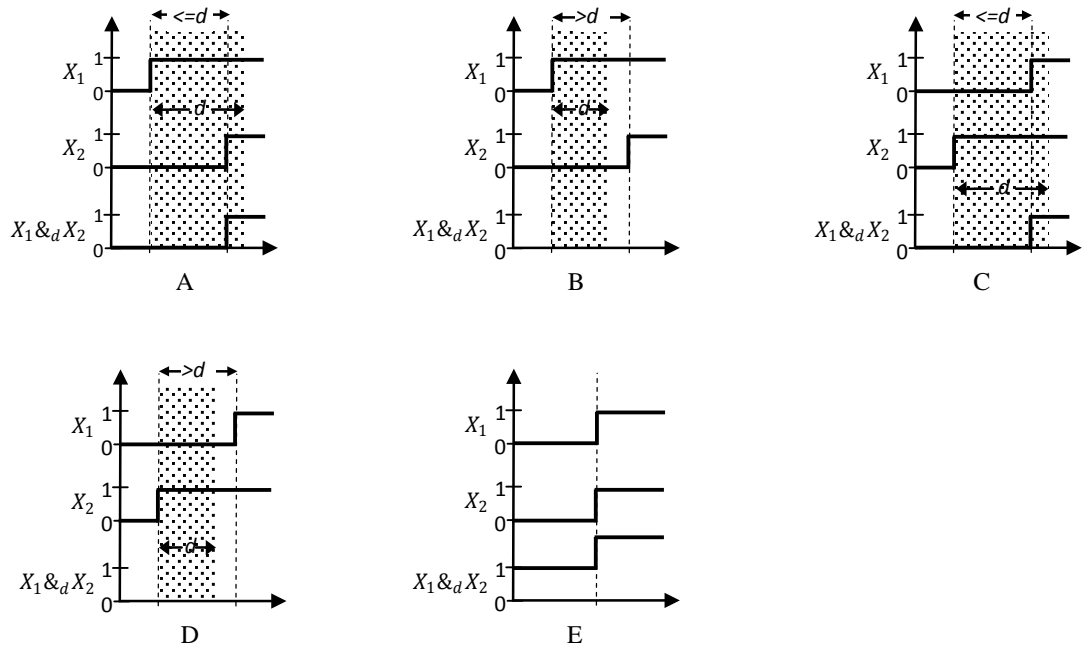


Figure 3.4-2: Timing models of a pSAND gate

The leftmost symbol ‘<’ in (3.4-2) and (3.4-3) is the less than symbol. In Fig. 3.4-2A X_1 occurs before X_2 within the duration that is less or equal to d but greater than zero; $X_1 \&_d X_2$ occurs. In Fig. 3.4-2B X_1 occurs before X_2 for a time interval greater than d ; $X_1 \&_d X_2$ does not occur. In Fig. 3.4-2C X_2 occurs before X_1 within a duration, which is less or equal to d but greater than zero; $X_1 \&_d X_2$ occurs.

In Fig. 3.4-2D X_2 occurs before X_1 within a time interval greater than d ; $X_1 \&_d X_2$ does not occur. In Fig. 3.4-2E X_1 occurs at exactly the same time as X_2 ; $X_1 \&_d X_2$ does occur though it is zero. The parameterized SAND occurs when all of its input events occur within a specified duration.

3.4.2 Analytical Model

This section contains discussions on the quantitative analysis of the pSAND gate using analytical techniques. This is done for MCSQs with only two events and a generic formula for MCSQs with n number of events.

pSAND Term with 2 Input Events

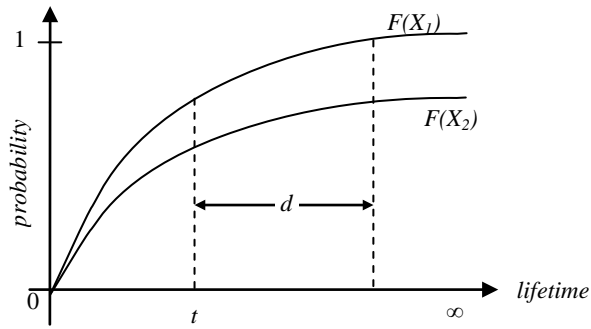


Figure 3.4-3: Graph of two nearly simultaneous events

From Fig. 3.4-3, the probability of X_1 and X_2 occurring between t and $t+d$ is the sum of the probabilities of X_1 and X_2 occurring at $t+d$ less the sum of the probabilities of X_1 and X_2 occurring at t . Mathematically,

$$\begin{aligned} P(X_1 \&_d X_2)\{t, t + d\} \\ &= [P(X_1 < X_2)\{t + d\} + P(X_2 < X_1)\{t + d\}] \\ &\quad - [P(X_1 < X_2)\{t\} + P(X_2 < X_1)\{t\}] \end{aligned}$$

From Pandora's Completion law in (3.3-6),

$$\begin{aligned} P(X_1.X_2)\{t\} &= P(X_1 < X_2)\{t\} + P(X_1 \&_d X_2)\{t\} + P(X_2 < X_1)\{t\} \\ P(X_1.X_2)\{t + d\} &= P(X_1 < X_2)\{t + d\} + P(X_1 \&_d X_2)\{t + d\} + P(X_2 < X_1)\{t + d\} \end{aligned}$$

Also, from the graph in Fig. 3.4-3, it is evident that

$$P(X_1.X_2)\{t + d\} = F(X_1)\{t\}.F(X_2)\{t + d\} + F(X_1)\{t + d\}.F(X_2)\{t\}$$

However,

$$\begin{aligned} F(X_1)\{t\}.F(X_2)\{t + d\} &= P(X_1 < X_2)\{t + d\} \\ F(X_1)\{t + d\}.F(X_2)\{t\} &= P(X_2 < X_1)\{t + d\} \end{aligned}$$

Thus,

$$P(X_1.X_2)\{t + d\} = P(X_1 < X_2)\{t + d\} + P(X_2 < X_1)\{t + d\}$$

And,

$$\begin{aligned}
P(X_1 \&_d X_2)\{t, t + d\} \\
&= [P(X_1 < X_2)\{t + d\} + P(X_2 < X_1)\{t + d\}] \\
&\quad - [P(X_1 < X_2)\{t\} + P(X_2 < X_1)\{t\}]
\end{aligned} \tag{3.4-7}$$

Where $d=0$, $\&_d = \&$. Substituting $d=0$ into (3.4-7),

$$\begin{aligned}
P(X_1 \&_d X_2)\{t, t + 0\} \\
&= [P(X_1 < X_2)\{t + 0\} + P(X_2 < X_1)\{t + 0\}] \\
&\quad - [P(X_1 < X_2)\{t\} + P(X_2 < X_1)\{t\}] \\
P(X_1 \&_d X_2)\{t, t + d\} &= 0
\end{aligned}$$

The SAND and PAND gates are specific instances of the pSAND gate. As d approaches zero, the pSAND value approaches the SAND value, which is zero. It is also evident that pSAND is commutative; $X_1 \&_d X_2$ is equal to $X_2 \&_d X_1$. Where d is zero, SAND is zero and iPAND is equal to ePAND.

pSAND Term with n Input Events

Where $F\{X\}(t)$ is the probability that X occurs any time before t and $F\{X\}(t_0, t_1)$ is the probability that X occurs between t_0 and t_1 , then,

For any two input events X_1 and X_2 ,

$$P(X_1 \&_d X_2)\{t, t + d\} = F(X_1)\{t\}.F(X_2)\{t, t + d\} + F(X_2)\{t\}.F(X_1)\{t, t + d\}$$

For any three input events X_1, X_2 and X_3 ,

$$\begin{aligned}
P(X_1 \&_d X_2 \&_d X_3)\{t, t + d\} \\
&= F(X_1)\{t\}.F(X_2)\{t, t + d\}.F(X_3)\{t, t + d\} \\
&\quad + F(X_2)\{t\}.F(X_1)\{t, t + d\}.F(X_3)\{t, t + d\} \\
&\quad + F(X_3)\{t\}.F(X_1)\{t, t + d\}.F(X_2)\{t, t + d\}
\end{aligned}$$

For any four input events X_1, X_2, X_3 and X_4 ,

$$\begin{aligned}
P(X_1 \&_d X_2 \&_d X_3 \&_d X_4)\{t, t + d\} \\
&= F(X_1)\{t\}.F(X_2)\{t, t + d\}.F(X_3)\{t, t + d\}.F(X_4)\{t, t + d\} \\
&\quad + F(X_2)\{t\}.F(X_1)\{t, t + d\}.F(X_3)\{t, t + d\}.F(X_4)\{t, t + d\} \\
&\quad + F(X_3)\{t\}.F(X_1)\{t, t + d\}.F(X_2)\{t, t + d\}.F(X_4)\{t, t + d\} \\
&\quad + F(X_4)\{t\}.F(X_1)\{t, t + d\}.F(X_2)\{t, t + d\}.F(X_3)\{t, t + d\}
\end{aligned}$$

For any n input events X_1, X_2, \dots, X_{n-1} and X_n ,

$$P(X_1 \&_d X_2 \&_d \dots \&_d X_{n-1} \&_d X_n) \{t_0, t_1\} \\ = \sum_{i=1}^n \left(F(X_i) \{t_0\} \cdot \left(\prod_{\substack{j=1 \\ j \neq i}}^n F(X_j) \{t_0, t_1\} \right) \right) \quad (3.4-8)$$

3.4.3 Monte Carlo Solution

The algorithm below is a generic Monte Carlo simulation for estimating the pSAND gate for n events. In the algorithm, a count, S , is kept of the number of times each event occurs at t while the remaining events occur between t and $t + d$. This is repeated for a large number of times, T , and dividing S by T produces the pSAND probability.

Algorithm 3.4-1: A Generic Monte Carlo simulation for the pSAND gate

Require: $X[n]$, d

$S \leftarrow 0$

for $i = 1$ to n **do**

$R[i] \leftarrow \text{NextRandomNumber}$

end for

for $k = 1$ to T **do**

for $i = 1$ to n **do**

for $j = 1$ to n **do**

if $(j \neq i)$

if $(R[i] \leq Pr(X[i])\{t\} \ \&\& \ R[j] > Pr(X[j])\{t\} \ \&\& \ R[j] \leq Pr(X[j])\{t + d\})$

then

$S \leftarrow S + 1$

end if

end if

end for

end for

end for

return S/T

3.5 MCSQ and Top-Event Quantification

A qualitative analysis of a fault tree produces cut sets, or cut sequences for a dynamic or temporal fault tree. Cut sequences are conjunctions of sequences, and the cut sequences themselves are connected as part of a disjunction — the occurrence of any one cut sequence will cause the top event to occur. Thus the most natural representation for cut sequences and MCSQs is in a sum-of-products form, also known as disjunctive normal

form. Quantitative analysis of Pandora fault trees follow a prior qualitative analysis. The qualitative analysis produces a set of MCSQs, which contains no redundancies and contradictions. Before quantitative analysis starts, pSAND gates are identified and assigned corresponding d values. We discuss below the evaluation of MCSQs with one type of gate and MCSQs with different gates.

3.5.1 MCSQ with One Type of Operator

This subsection discusses the evaluation of MCSQs which contains two or more different input events but only one type of operator; the operator may repeat more than once. Evaluating a MCSQs with one type of operator qualitatively is simple but if not meticulously done can be wrongly evaluated. For this reason, we discuss MCSQs with each gate below. SAND gates evaluate to zero so they are ignored.

For MCSQs with more than one operator in a particular sequence, the leftmost operator has priority over its immediate right operator. Given that ‘*’ is any of the operators in Pandora,

$$P(X_1 * X_2 * \dots * X_{n-1} * X_n)\{t\} = P\left(\left(\left(\left(X_1 * X_2\right) * \dots * X_{n-1}\right) * X_n\right)\right)\{t\}$$

For example,

$$P(X_1 < X_2 < \dots < X_{n-1} < X_n)\{t\} = P\left(\left(\left(\left(X_1 < X_2\right) < \dots < X_{n-1}\right) < X_n\right)\right)\{t\}$$

The input events X_1, X_2, \dots, X_n of a MCSQ could be basic events or intermediate events. The type of input event is not necessary here because, depending on the level of abstraction, an intermediate event can be an input event. If it is a basic event, $P(X_i)\{t\} = F(X_i)\{t\}$. However, if it is an intermediate event, $P(X_i)\{t\}$ will have to be evaluated on the input events of X_i . The focus here is the wrong evaluation of a particular $P(X_i)\{t\}$ by replacing $F(X_i)\{t\}$ with λ_i or vice versa. Some of the formulae are expressed in terms of failure rates, λ , whilst others are expressed in probabilities P . For the sake of uniformity, we describe all the gates in Pandora using failure rates expressions.

Recalling the AND gate formula for a MCSQ of n input events from (2.2-19)

$$\begin{aligned} P(MCSQ)\{t\} &= P(X_1)\{t\}.P(X_2)\{t\}.\dots.P(X_{n-1})\{t\}.P(X_n)\{t\} \\ &= (1 - e^{-\lambda_1 t}).(1 - e^{-\lambda_2 t}).\dots.(1 - e^{-\lambda_{n-1} t}).(1 - e^{-\lambda_n t}) \end{aligned} \quad (3.5-1)$$

The OR formula using Esary-Proschan (2-2-23) for a MCSQ of n input events is

$$\begin{aligned} P(MCSQ)\{t\} &= 1 - \left((1 - P(X_1)) \cdot (1 - P(X_1)) \cdot \dots \cdot (1 - P(X_n)) \right) \\ &= 1 - (e^{-\lambda_1 t} \cdot e^{-\lambda_2 t} \cdot \dots \cdot e^{-\lambda_{n-1} t} \cdot e^{-\lambda_n t}) \end{aligned} \quad (3.5-2)$$

Describing the pSAND expression in (3.4-8) in terms of failure rate is

$$P(MCSQ)\{t_0, t_1\} = \sum_{i=1}^n \left((1 - e^{-\lambda_i t_0}) \cdot \left(\prod_{\substack{j=1 \\ j \neq i}}^n (1 - e^{-\lambda_j (t_1 - t_0)}) \right) \right) \quad (3.5-3)$$

POR (3.1-33) and PAND (3.2-5) formulae are already expressed in terms of failure rate. For a reminder, they are stated below as (3.5-4) and (3.5-5) respectively.

$$P(MCSQ)\{t\} = \frac{\lambda_1 \left(1 - (e^{-\sum_{i=1}^n \lambda_i t}) \right)}{\sum_{i=1}^n \lambda_i} \quad (3.5-4)$$

$$P(MCSQ)\{t\} = \prod_{i=1}^n \lambda_i \sum_{k=0}^n \left[\frac{e^{(a_k t)}}{\prod_{\substack{j=0 \\ j \neq k}}^n (a_k - a_j)} \right] \quad (3.5-5)$$

Where $a_0 = 0$ and $a_m = -\sum_{j=1}^m \lambda_j$ for $m > 0$.

(3.5-1), (3.5-2), (3.5-3), (3.5-4) and (3.5-5) are analytical techniques for evaluating MCSQs with only one type of gate whilst algorithms 2.2-1, 2.2-2, 3.1-1, 3.2-1 and 3.4-1 are Monte Carlo simulations for evaluating such MCSQs.

3.5.2 MCSQ with Different Operators

In this thesis, the quantification of basic or intermediate events with Boolean logical operators AND and OR have been reviewed and various techniques for the POR, PAND and pSAND gates have been formulated. It must also be noted that a operator precedence is required for all temporal and Boolean gates in Pandora before quantitative analysis can take place. The precedence of operator evaluation in Pandora is:

$$\text{SAND} \rightarrow \text{pSAND} \rightarrow \text{PAND} \rightarrow \text{POR} \rightarrow \text{AND} \rightarrow \text{OR}$$

Where the precedence is in descending order; thus SAND has higher precedence than PAND and $X \& Y < Z$ should be evaluated as $(X \& Y) < Z$ and,

$$P(G.A < B|(C + D.F\&E))\{t\} = P\left(\left(G.\left((A < B)|(C + (D.(F\&E)))\right)\right)\right)\{t\}$$

So far, straight forward generic Monte Carlo simulations have been constructed for evaluating MCSQs with only one type of gate. However, MCSQs with combinations of different temporal gates require more effort to construct. To construct Monte Carlo simulations for some of these MCSQs, the MCSQs are first categorised into groups. Most MCSQs containing any of the temporal gates, |, <, and/or & will usually have | succeeding < and &. For example $A < B|C$, $A \&_d B|C$, $A \&_d B < C$, $A < B \&_d C$ are MCSQs but $A|B \&_d C$, $A|B < C$ are not. To demonstrate the scale of these complexities and how they can be solved, consider a top event $T = A < B|C \cdot E + A < B \&_d E \cdot C$. Before constructing a Monte Carlo simulation for T , we discuss the conditions necessary for achieving its MCSQs: $A < B|C \cdot E$ and $A < B \&_d E \cdot C$.

To model $A < B|C \cdot E$, it must be noted that $A < B|C \cdot E = ((A < B)|C) \cdot E$. The time-to-failure, $TTF(x, r)\{t\}$, and CDF, $F(x)\{t\}$, of all events will be required (where x is the failure rate of an event and r is the random number to simulate the probability of that event). As mentioned earlier, the condition required for $A < B$, $A \text{ and } B$, to occur, if && is the logical AND, is

$$\begin{aligned} A \text{ and } B &= R_A \leq F(\lambda_A)\{t\} \&\& R_B \leq F(\lambda_B)\{t\} \&\& TTF(\lambda_A, R_A)\{t\} \\ &< TTF(\lambda_B, R_B)\{t\} \end{aligned}$$

Which means the simulated instance of A occurs, that of B also occurs and the next time-to-failure of A is less than that of B . Evaluating $A < B$ is quite simple. However, this cannot be said of $A < B|C$. Assume $A < B$ evaluates to X , then the condition necessary for $X|C$, $X \text{ por } C$, to occur, if || is the logical OR, is

$$\begin{aligned} X \text{ por } C &= (R_X \leq F(\lambda_X)\{t\} \&\& R_C \leq F(\lambda_C)\{t\} \&\& TTF(\lambda_X, R_X)\{t\} \\ &< TTF(\lambda_C, R_C)\{t\}) || (R_X \leq F(\lambda_X)\{t\} \&\& R_C > F(\lambda_C)\{t\}) \end{aligned}$$

Which means the simulated instances of X and C occur and X occurs before C or X occurs but C does not. The next step is the incorporation of $A \text{ and } B$ into $X \text{ por } C$. This presents some challenges. $A \text{ and } B$ is a Boolean value which could be a True or a False. Evaluating $F(\lambda_X)\{t\}$ or $TTF(\lambda_X, R_X)\{t\} < TTF(\lambda_C, R_C)\{t\}$ requires numerical values

(failure rates) not Boolean values. Therefore, simply replacing X with $A\text{pand}B$ is impractical.

To solve this problem, one must consider the sequential order in which the events A , B and C occur. $A<B$ means A occurs before B . $A<B|C$ means A occurs before B and B occurs before C if C should occur. Meaning $TTF(\lambda_X, R_X)\{t\} < TTF(\lambda_C, R_C)\{t\}$ can be replaced by $TTF(\lambda_B, R_B)\{t\} < TTF(\lambda_C, R_C)\{t\}$. Therefore, the condition required for $A<B|C$, $A\text{pand}B\text{por}C$, to occur is

$$\begin{aligned} A\text{pand}B\text{por}C &= (A\text{pand}B \ \&\& \ R_C \leq F(\lambda_C)\{t\} \ \&\& \ TTF(\lambda_B, R_B)\{t\} \\ &< TTF(\lambda_C, R_C)\{t\}) \ || \ (A\text{pand}B \ \&\& \ R_C > F(\lambda_C)\{t\}) \end{aligned}$$

And the condition required for the first MCSQ, $A<B|C \cdot E$, $A\text{pand}B\text{por}C\text{and}E$ is

$$\begin{aligned} A\text{pand}B\text{por}C\text{and}E &= (A\text{pand}B \ \&\& \ R_C \leq F(\lambda_C)\{t\} \ \&\& \ TTF(\lambda_B, R_B)\{t\} \\ &< TTF(\lambda_C, R_C)\{t\}) \ || \ (A\text{pand}B \ \&\& \ R_C > F(\lambda_C)\{t\}) \ \&\& \ R_E \\ &\leq F(\lambda_E)\{t\} \end{aligned}$$

The second MCSQ, $A<B\&_dE \cdot C$, follows a similar approach however, differs slightly. Firstly, it must be noted that, $A<B\&_dE \cdot C = (A<(B\&_dE)).C$. The condition required for $B\&_dE$ occurring is

$$\begin{aligned} B\text{sand}E &= R_B \leq F(\lambda_B)\{t\} \ \&\& \ R_E > F(\lambda_E)\{t\} \ \&\& \ R_E \leq F(\lambda_E)\{t + d\} \ || \ R_E \\ &\leq F(\lambda_E)\{t\} \ \&\& \ R_B > F(\lambda_B)\{t\} \ \&\& \ R_B \leq F(\lambda_B)\{t + d\} \end{aligned}$$

The next level to consider is $A<B\&_dE$. Again assuming $B\&_dE=X$, the condition required for $A<X$, to occur has already been stated above. Again, it is impractical to substitute $B\text{sand}E$ into into $A<X$. A logical solution to this challenge would be to evaluate $A<B\&_dE$ using for following condition

$$\begin{aligned} A\text{pand}B\text{sand}E &= R_A \leq F(\lambda_A)\{t\} \ \&\& \ B\text{sand}E \ \&\& \ TTF(\lambda_A, R_A)\{t\} \\ &< TTF(\lambda_B, R_B)\{t\} \ \&\& \ TTF(\lambda_A, R_A)\{t\} < TTF(\lambda_E, R_E)\{t\} \end{aligned}$$

Meaning, A occurs before the simultaneous occurrence of B and E . Finally, the condition required for $A<B\&_dE \cdot C$ occurring is

$$\begin{aligned} A\text{pand}B\text{sand}E\text{and}C &= R_A \leq F(\lambda_A)\{t\} \ \&\& \ B\text{sand}E \ \&\& \ TTF(\lambda_A, R_A)\{t\} \\ &< TTF(\lambda_B, R_B)\{t\} \ \&\& \ TTF(\lambda_A, R_A)\{t\} \\ &< TTF(\lambda_E, R_E)\{t\} \ \&\& \ R_C \leq F(\lambda_C)\{t\} \end{aligned}$$

So far, the Monte Carlo conditions required for MCSQs with two temporal gates have been constructed. Unfortunately, each MCSQ in a Monte Carlo simulation, unlike the analytical solution, needs to be modelled individually. For example, to evaluate $A < B \cdot C \&_d E | F$ analytically, generic formulae for evaluating pSAND, POR, PAND and AND could be used to evaluate $(A < B) \cdot ((C \&_d E) | F)$. However, for a Monte Carlo solution, $A < B \cdot C \&_d E | F$ will have to be constructed and not evaluated with a generic formula. For this reason, there is need for the individual construction of MCSQs such as $A < B \& C | D \cdot E$ etc.

Appendix 1 contains a list Monte Carlo conditions for some MCSQs with four or less events. It contains MCSQs with combinations of events and temporal gates only. MCSQs containing static gates – AND and OR – are ignored because:

1. If an AND gate is present in a MCSQ, a simple logical AND is needed for their modelling. For example, in the construction of $A < B \&_d C \cdot E$ above, a logical AND condition, $\& \& R_E \leq F(\lambda_E)\{t\}$, was appended to the condition of $A < B \&_d C$.
2. It is rare to find an OR gate in one particular MCSQ; OR gates are usually logical disjunctions between various MCSQs. If a MCSQ contains an OR gate, a simple logical OR can be used to construct the condition required for it occurring.

Full quantitative analysis of temporal fault trees of safety-critical system can now be evaluated. Quantitative analysis of real world critical systems containing several hundreds of events and gates manually is almost impractical. Appendix 2 contains algorithms based on a modifications of the Shunting-Yard Algorithm (Dijkstra 1961) and the Reverse Polish Notation (Burks *et al.* 1954; Hamblin 1962) to provide a programmatic solution which takes in a string of MCSQs, parses them into appropriate RPN, evaluates them and calculates the top-event probability.

3.6 Evaluating Systems with Different Failure Distributions

So far in this thesis, the analytical techniques and simulations proposed are restricted to the exponential distribution. In real world scenarios, a safety-critical system may feature many components with different failure distributions. The Weibull and lognormal distributions are common failure distributions associated with many components

(O'Connor 2011). We present algorithms for modelling, simulating and evaluating such dynamic systems with Monte Carlo simulation.

All temporal gates used in Pandora have particular sequences. Therefore to model these gates, one would have to consider the sequential dependencies encapsulated by these gates. Just as earlier described, the TTF property of events is use to model the dynamic behaviours of temporal gates. As already mentioned, simulation for the exponential distributions have already been provided for the PAND (Algorithm 3.2-1), POR (Algorithm 3.1-1), pSAND (Algorithm 3.4-1), OR (Algorithm 2.2-1) and AND (Algorithm 2.2-2) gates.

We present the modelling of all these gates for both the Weibull and Lognormal distributions. It must be noted that throughout this section, $F(a, b, t)$ and $TTF(a, b, r, t)$, are the CDF and TTF respectively for an event with $\alpha=a$ and $\beta=b$ (for a Weibull distribution) or $\mu=a$ and $\sigma=b$ (for a Lognormal distribution); r is a random number representing the failure probability of an event. r_x means the failure probability of an even X . 't' represents the system lifetime. The TTF of the Weibull and lognormal distributions below, can be calculated from the CDF of both distributions stated in (2.1-22) and (2.1-18) respectively.

$$TTF_{Weibull} = \alpha \cdot \sqrt[\beta]{\log\left(\frac{1}{1-r}\right)} \quad (3.6-1)$$

$$TTF_{lognormal} = e^{\sigma \cdot \Phi^{-1}(r) + \mu} \quad (3.6-2)$$

We provide algorithms for modelling the pSAND, PAND and POR gates conditions for Monte Carlo simulation of two events x and y below. $\&\&$ and $\|$ represent the logical AND and OR respectively.

Algorithm 3.6-1: CDF of event X having a Weibull distribution

Require: a, b, r_x, t

return $r_x \leq (1 - \text{Exp}(-\text{Power}(t/a, b)))$

Algorithm 3.6-2: CDF of event X having a Lognormal Distribution

Require: a, b, r_x, t

return $r_x \leq \text{CDF}(\text{Log}(t) - a) / b$ //Note: CDF is the standard normal CDF function

Algorithm 3.6-3: TTF for Weibull Distribution

Require: a, b, r, t

return $a * \text{Power}(\text{Log}(1 / (1 - r)), 1 / b)$

Algorithm 3.6-4: TTF for Lognormal Distribution**Require:** a, b, r, t **return** $Exp(b * InverseCDF(r) + a)$ **Algorithm 3.6-5: SimPAND****Require:** $r_x, x_a, x_b, r_y, y_a, y_b, t$ **return** $r_x \leq F(x_a, x_b, t) \ \&\& \ r_y \leq F(y_a, y_b, t) \ \&\& \ TTF(x_a, x_b, r_x, t) < TTF(y_a, y_b, r_y, t)$ **Algorithm 3.6-6: SimPOR****Require:** $r_x, x_a, x_b, r_y, y_a, y_b, t$ **return** $(r_x \leq F(x_a, x_b, t)) \ \&\& \ (r_y \leq F(y_a, y_b, t)) \ \&\& \ TTF(x_a, x_b, r_x, t) < TTF(y_a, y_b, r_y, t) \ \|\ (r_x \leq F(x_a, x_b, t)) \ \&\& \ (r_y > F(y_a, y_b, t))$ **Algorithm 3.6-7: SimSAND****Require:** $r_x, x_a, x_b, r_y, y_a, y_b, d, t$ **return** $(r_x \leq F(x_a, x_b, t) \ \&\& \ r_y > F(y_a, y_b, t) \ \&\& \ r_y \leq F(y_a, y_b, t+d)) \ \|\ (r_y \leq F(y_a, y_b, t) \ \&\& \ r_x > F(x_a, x_b, t) \ \&\& \ r_x \leq F(x_a, x_b, t+d))$

SimPAND, SimSAND and SimPOR are generic conditions for modelling the PAND, pSAND and POR conditions for Monte Carlo simulation for both the Weibull and lognormal distributions. To demonstrate how to model a system with different failure distributions, we consider the HPS example described in chapter 1. The following assumptions are made:

1. the medical device, MD , has a cumulative damage effect and therefore has an exponentially distributed failure behaviour.
2. the main electricity sub-system, E , is subject to corrosion maintenance and therefore has a failure behaviour of a Weibull distribution.
3. the generator sub-system, G , is subject to a repair time distribution that is a lognormal distribution.
4. MD fails if E and S fail 0.05 seconds apart; that is $d = 0.05$.
5. the sensor, S , comprises of capacitors and integrated circuits with exponential distribution.
6. The input command to power the medical device, I , is trivial; the input will always be made when necessary.

As a reminder, the MCSQs of HPS are:

1. I (Input command – this is ignored from hence forth)
2. MD (internal failure of medical device)
3. $S < E$ (failure of sensor before electricity)
4. $S \&_{0.05} E$ (simultaneous failure of sensor and electricity)
5. $E.G$ (failure of electricity and generator)

Before modelling the entire system failure, we construct simulation conditions that must be satisfied for each MCSQs to occur.

Algorithm 3.6-8: SimMD

Require: r_{md} , md , t

return $r \leq (1 - \text{Exp}(-md*t))$

In Algorithm 3.6-8 r_{md} , and t as as described earlier. md is the failure rate of MD . The algorithm determines if MD , which is exponentially distributed, has occurred.

Algorithm 3.6-9: SimSpandE

Require: r_s , s , r_e , e_a , e_b , t

return $r_s \leq (1 - \text{Exp}(-s*t)) \&\& r_e \leq (1 - \text{Exp}(-\text{Power}((t/e_a), e_b))) \&\& (1/s) * \text{Log}(1/(1 - r_s)) < e_a * \text{Power}(\text{Log}(1/(1 - r_e)), 1/e_b)$

Algorithm 3.6-9 models the condition where S occurs before $E - S < E$. $S < E$ occurs when S occurs and E occurs and the TTF of S is less than that of E . In this model, S is exponentially distributed and E has a Weibull distribution.

Algorithm 3.6-10: SimSsandE

Require: r_s , s , r_e , e_a , e_b , d , t

return $(r_s \leq (1 - \text{Exp}(-s*t)) \&\& r_e > (1 - \text{Exp}(-\text{Power}((t/e_a), e_b))) \&\& r_e \leq (1 - \text{Exp}(-\text{Power}((t+d/e_a), e_b)))) \&\& (r_e \leq (1 - \text{Exp}(-\text{Power}((t/e_a), e_b))) \&\& r_s > (1 - \text{Exp}(-s*t)) \&\& r_s \leq (1 - \text{Exp}(-s*(t+d))))$

Algorithm 3.6-10 models the condition where S and E occur within a duration of interval $d - S \&_d E$. S is exponentially distributed and E has a Weibull distribution.

Algorithm 3.6-11: SimEandG

Require: r_e , e_a , e_b , r_g , g_a , g_b , t

return $(r_e \leq (1 - \text{Exp}(-\text{Power}((t/e_a), e_b))) \&\& r_g \leq (\text{CDF}(\text{Log}(t) - g_a) / g_b)$

Algorithm 3.6-11 is a simple condition that checks if E and G have occurred; the former has Weibull failure distribution whilst the later has lognormal failure distribution.

Algorithm 3.6-12: Estimating the HPS system failure probability

$S \leftarrow 0$

for $k = 1$ to T **do**

if ($SimMD$ // $SimSpandE$ // $SimSsandE$ // $SimEandG$) **then**

$S \leftarrow S + 1$

end if

return S/T

Algorithm 3.6-12 is a Monte Carlo simulation model that uses the previously discussed conditions to estimate the probability of a total system failure of HPS. These different component/sub-system failure distributions of a particular system are rarely discussed in literature; most researchers assume a system to have one failure distribution.

Chapter Four

TEMPORAL QUALITATIVE ANALYSIS

The primary goal of this thesis is the quantitative analysis of temporal fault trees of safety-critical systems. This chapter provides two new improvements to the qualitative analysis that makes the quantitative analysis possible. First of all, it describes how a temporal safety analysis technique – Pandora – can be modularised to enhance its efficiency. The modularisation approach proposed is a simple technique that reduces the size of cut sequences to be analysed at a particular instance by breaking the cut sequences into logical chunks that can be analysed recursively. The second technique is based on Binary Decision Diagrams (BDD), where temporal behaviours are introduced into BDDs and evaluated with novel procedures to produce MCSQs from which quantitative analysis can be performed. Finally, some examples are given to demonstrate the potential of these techniques.

4.1 Groups and Modules Modularization Technique (GMMT)

In fault tree analysis, modularisation is, primarily, a way of categorizing large fault trees into logically related modules to enhance computational efficiency (Dutuit and Rauzy 1996). In DFTs, modularization is used to group dynamic fault trees into those with temporal gates and those with static gates. Modules with static gates are analysed using traditional FTA approaches, especially BDD, whilst modules with dynamic gates are analysed with Markov approaches (Gulati and Dugan, 1997). This modular approach harnesses the strengths of both techniques. By doing so, the computational efficiency and accuracy of the evaluations are enhanced.

As mentioned earlier, Pandora, a temporal fault tree analytic technique, has two primary approaches – Archimedes and Euripides – for logically analysing temporal fault trees. The third approach is a combination of both techniques and the use of modularisation (Walker, 2009). Pandora employs the Linear Time Algorithm (LTA) (Dutuit and Rauzy, 1996) for evaluating large fault trees. LTA, unlike Gulati and Dugan's (1997) technique, focuses on 'independent' subtrees of a large fault tree. A simple example of how LTA can be used in Pandora is the evaluation of the expression $A \& (B|D + B.D) + (C|(E \bullet (F+E)))$ described in (Walker, 2009).

Using LTA, this expression has four modules:

Module 1: $A \& (B/D + B.D)$

Module 2: $(B/D + B.D)$

Module 3: $(C/(E.(F+E)))$

Module 4: $(E.(F+E))$

The fourth module has only static gates and is analyzed with classical FTA techniques to produce E . Substituting this result into the third module produces $C|E$. Archimedes or Euripides could be used in evaluating the second module and this will produce B . Substituting this into the first module produces $A \& B$. Therefore the MCSQ of the expression is $A \& B + C|E$. Unfortunately, this technique may not be able to detect some subtle completions that may exist in CSQs such as:

$$A \& (B/D + B.D) + (A \langle (B.(F+B))) + (B \langle A).(A/E + A.E)$$

Breaking the above CSQs into modules:

Module 1: $A \& (B/D + B.D)$

Module 2: $B/D + B.D = B$

Module 3: $(A \langle (B.(F+B)))$

Module 4: $B.(F+B) = B$

Module 5: $(B \langle A).(A/E + A.E)$

Module 6: $A/E + A.E = A$

Note that: $(B \langle A).A = A$

Substituting Module 6 into 5 produces $B \langle A$, 4 into 3 produces $A \langle B$ and 2 into 1 produces $A \& B$. Therefore the entire expression reduces to $A \& B + A \langle B + B \langle A$. Without applying Archimedes to these CSQs, one may think they are minimal; this is inaccurate. $A \& B + A \langle B + B \langle A$ actually reduces into $A \bullet B$. How then can such a set of CSQs be modularised so that such completions can be identified and solved? The quest to solve this challenge has led to the Groups and Modules (GM) modularisation technique.

The GM approach seeks to modularise temporal fault trees using Euripides and eliminating redundancies, contradictions and completions using Archimedes. GM has three principal phases: Classification, Combination, Comparison. The following are discussions of these phases in details.

4.1.1 Classification

Classification is the process of putting CSQs into groups and modules. A group is a set of CSQs with the same number of distinct events whilst a module is a set of CSQs with

exactly the same events. Classification has four processes: Euripides (already described), cut-off, grouping and regrouping.

Cut-Off

The cut-off stage applies a logical cut-off on CSQs. A cut-off value of 4 means only CSQs with a maximum of 4 distinct events will be considered in an analysis. That is to say, if a cut-off of 4 applies, $A \bullet B < A | C \& A \bullet D$ will not be cut off but $A \bullet B < C | D \& E$ will be cut off; the former has 4 distinct events but the later has 5.

Grouping

This process involves the arrangements of CSQs into groups; all CSQs in a group contain the same number of distinct events in increasing order. For example, Group 1 will contain only CSQs with one distinct event, Group 2 will contain CSQs with only two distinct events, Group 3 will contain CSQs with only 3 distinct events and so on. It does not matter the number of times an event repeats itself in a CSQs. Grouping does not count the total number of events but the number of distinct events. Therefore the CSQs, $A \bullet B < A | C \& A$, $A \bullet C | G$ and $X < Y < Z$, contain only three distinct events each and would be in the same group.

The final process of grouping is the classification of CSQs in groups into Modules. Modules contain exactly the same distinct events. Individual CSQs in a group are called Modules. Therefore, $A \bullet B < A | C \& A$ and $A < B < C$ will belong to the same module because they have exactly the same events – A , B and C – whilst $A \bullet C | G$ will belong to another module because though it contains A and B , it has G instead of a C . Both modules will however belong to the same group. To clarify the GM technique more practically, consider the ‘unminimised’ top-event below:

$$F/G + C + A < B + E \& B/D + A + A \& B + E < G + F.G + H \& D < G.I + A < B \& C + G < E + E \& G + E < B \& C + B.D$$

These CSQs can be arranged into four groups with modules as seen in the table below. Group 1 has two modules A and C . Group 2 has four modules, Group 3 has three modules and Group 4 has one module. Module 3 of Group 2, $G < E + E < G + E \& G$, has only two distinct events E and G though it has three CSQs. This table of groups (columns) with corresponding modules (rows) is referred to as the the Groups and Modules Table (GMT). GXY means Group X Module Y . Therefore, $G2M2$ is $B \bullet D$.

	<i>Group 1</i>	<i>Group 2</i>	<i>Group 3</i>	<i>Group 4</i>
<i>Module 1:</i>	<i>A</i>	<i>A&B + A<B</i>	<i>A<B&C</i>	<i>H&D<G.I</i>
<i>Module 2:</i>	<i>C</i>	<i>B.D</i>	<i>E&B/D</i>	
<i>Module 3:</i>		<i>G<E + E<G + E&G</i>	<i>E<B&C</i>	
<i>Module 4:</i>		<i>F/G + F.G</i>		

Regrouping

Once groups with modules have been formed, each module in the group is minimised using Archimedes. This is done from the first module in the first group to the last module in the last group. As each module is minimized, its number of distinct events can either decrease or remain the same; it cannot increase. If the number of events in the module decreases after this process, the module is regrouped – put in the group containing its number of distinct events. After a module is regrouped, it may belong to an entirely new module if none of the modules in the new group has the same distinct events or it may be added to a module with exactly the same events. The new module is also minimised and regrouped. This goes on until no regrouping is possible after a minimization.

From the example given in the Grouping phase, no module in Group 1 can be regrouped because they are critical events and cannot be minimised any further. G2M1 and G2M2 cannot also be minimised so they remain as they are. G2M3, $G<E + E<G + E&G$ minimizes to $G\bullet E$ but cannot be regrouped so it remains in Group 2. However, Module 4 $F|G + F\bullet G$ reduces to F therefore will be regrouped into Group 1 resulting in the GMT below.

	<i>Group 1</i>	<i>Group 2</i>	<i>Group 3</i>	<i>Group 4</i>
<i>Module 1:</i>	<i>A</i>	<i>A&B + A<B</i>	<i>A<B&C</i>	<i>H&D<G.I</i>
<i>Module 2:</i>	<i>C</i>	<i>B.D</i>	<i>E&B/D</i>	
<i>Module 3:</i>	<i>F</i>	<i>G.E</i>	<i>E<B&C</i>	

4.1.2 Combination

Combination is a recursive process of combining various CSQs and minimizing them using Archimedes. This is done by removing modules from their groups/modules, minimizing them and regrouping the result. The process starts by considering each module in turn, starting from the first group to the last. Each module is combined with other modules which contains all of its events, minimized and regrouped. Considering the example used in the classification phase, G1M1 cannot be combined with G1M2 or G1M3 because they belong to the same group, neither can it be combined with G2M2 because it does not contain A. The first possible combination is G1M1 with G2M1 – $A +$

$A \& B + A < B$ – because G2M1 contains all the events in G1M1. This should produce the result A , which when regrouped produces the following GMT.

	<i>Group 1</i>	<i>Group 2</i>	<i>Group 3</i>	<i>Group 4</i>
<i>Module 1:</i>	A	$B \cdot D$	$A < B \& C$	$H \& D < G \cdot I$
<i>Module 2:</i>	C	$G \cdot E$	$E \& B / D$	
<i>Module 3:</i>	F		$E < B \& C$	

With no modules in Group 2 containing all the events in G1M1, the next step in combination is to evaluate G1M2 with G3M1 – $A + A < B \& C$. This also produces A , therefore the resulting GMT becomes

	<i>Group 1</i>	<i>Group 2</i>	<i>Group 3</i>	<i>Group 4</i>
<i>Module 1:</i>	A	$B \cdot D$	$E \& B / D$	$H \& D < G \cdot I$
<i>Module 2:</i>	C	$G \cdot E$	$E < B \& C$	
<i>Module 3:</i>	F			

This ends the combination process with G1M1 because no other module in Group 3 or Group 4 contains all of its events. The next module in Group 1, G1M2, is considered for combination. No module in Group 2 contains all the events of G1M2 so no combination with Group 2 is possible. The only module in Group 3 containing the events of G1M2 is G3M2, $E < B \& C$; Evaluating G1M2 with G3M2, $C + E < B \& C$, produces C which is regrouped. The only module in Group 4 does not contain the events in G1M2 and the GM table reduces to

	<i>Group 1</i>	<i>Group 2</i>	<i>Group 3</i>	<i>Group 4</i>
<i>Module 1:</i>	A	$B \cdot D$	$E \& B / D$	$H \& D < G \cdot I$
<i>Module 2:</i>	C	$G \cdot E$		
<i>Module 3:</i>	F			

The next step is the combination of F in G1M3 but there is no module in groups 2, 3 and 4 containing F so the GM table remains unchanged. With all modules in Group 1 considered, modules in Group 2 will be used for combination. Combination does not consider groups of lower levels, therefore, the first module to be reduced is $E \& B / D$ in Group 3. $B \cdot D + E \& B / D$ produces $B \cdot D + B \& E$. It must be noted that before minimization, both $B \cdot D$ and $E \& B / D$ are removed from their respective groups and after minimization the result, $B \cdot D$ and $B \& E$ is regrouped producing the GMT below.

	<i>Group 1</i>	<i>Group 2</i>	<i>Group 3</i>	<i>Group 4</i>
<i>Module 1:</i>	A	$G \cdot E$		$H \& D < G \cdot I$
<i>Module 2:</i>	C	$B \cdot D$		
<i>Module 3:</i>	F	$B \& E$		

At this point, Group 3 becomes void of modules. We continue to reduce $H\&D\<G.F$ in Group 4 using $B\bullet D$. Unfortunately, $H\&D\<G\bullet I$ does not contain all the distinct events in $B\bullet D$. For the same reason $G\bullet E$ and $B\&E$, the remaining modules in Group 3 cannot be used for combination with $H\&D\<G\bullet I$. Since Group 3 has no modules, the combination process ends.

4.1.3 Comparison

Comparison is a simple phase that has three processes – declassifying, classification and comparison. The declassification is the process of converting GMTs into a set of CSQs. Classification has already been described. The final process is the comparison, where the GMT from the classification process in this phase is compared to the previous GMT in the combination phase. If both GMTs are not the same Combination and Comparison are repeated until they are. When they are the same, the modules in the final GMT are the MCSQs. Applying these processes to the previous GMT will involve the following:

Declassification - converting GMT into CSQs:

$$A + C + F + B.D + G.E + B\&E + H\&D\<G.I$$

Classifying the result:

	<i>Group 1</i>	<i>Group 2</i>	<i>Group 3</i>	<i>Group 4</i>
<i>Module 1:</i>	<i>A</i>	<i>G.E</i>		<i>H\&D\<G.I</i>
<i>Module 2:</i>	<i>C</i>	<i>B.D</i>		
<i>Module 3:</i>	<i>F</i>	<i>B\&E</i>		

The final step is comparison. Comparing the above GMT with that from the combination stage shows that the GMT produced in both phases are the same. This completes the GM analysis and the various modules $A, C, F, B\bullet D, G\bullet E, B\&E, H\&D\<G\bullet I$ are the MCSQs.

4.1.4 Strengths and Limitations of the GMMT Approach

The GMMT approach, unlike Euripides, is able to detect and eliminate subtle completions. This is due to the fact that it evaluates all CSQs with the same number of distinct events in one module using Archimedes. Also, unlike Archimedes, GMMT is not restricted by the number of distinct events provided the cut-off is less than 4. In the example used in section 4.1.3, nine distinct events, $A, B, C, D, E, F, G, H, I$, were present in the analysis.

Using Archimedes alone to evaluate a set of CSQs with nine distinct events will require high computing resources and time (evaluating $2.96917108E10$ possible temporal sequences); it is certainly impractical to do this manually. However, the GMMT approach is such that Archimedes is used several times to evaluate CSQs of smaller, usually less than 4, distinct events.

Unfortunately, GMMT has some limitations. Though it can be used for CSQs with several distinct events, due to its use of Archimedes, it is restricted to a cut-off of 4. Meaning, no single CSQs should have more than 4 distinct events. This may not be a limitation in some cases where a logical cut-off of 4 is required; CSQs with more than four distinct events are less probable to occur. Another disadvantage is the fact that Archimedes is used several times in GMMT. An ideal solution will be to have Archimedes evaluate the entire set of CSQs in one evaluation. However, at the moment, this is a challenge when CSQs with more than 5 distinct events are considered.

Finally, GMMT may not be very practical until an efficient technique for analyzing Archimedes and/or Euripides is developed. However, until then, it tremendously improves the qualitative analysis of temporal fault trees using both Archimedes and Euripides. A more realistic application of GMMT is described in Chapter 5.

4.2 Temporal Binary Decision Diagrams (TBDD)

As discussed in earlier chapters, BDD is an alternative technique for evaluating traditional fault trees. Previous research (Sinnamon and Andrews 1998) shows that with an appropriate event ordering, they can accurately and efficiently evaluate the top-event probability without minimising the fault tree. BDDs are mainly employed in classical FTA. The author is not aware of any use of BDDs in temporal fault trees.

In this chapter, a novel technique for analysing temporal fault trees using BDDs is discussed. This technique, known as Temporal Binary Decision Diagram (TBDD), uses a mathematical sub-tree containment technique in minimising temporal fault trees to produce MCSQs which are then used in evaluating the top-event probability. Note that, unlike the classical BDD, TBDD is unable to produce the top-event probability from 'unminimised' temporal fault trees but is useful because it forms the foundation for evaluating temporal fault trees with BDDs.

4.2.1 Introducing Temporal Binary Decision Diagrams

This section contains a novel technique for analysing temporal fault trees using Binary Decision Diagrams (BDDs). It commences with the definition of a modified If-Then-Else structure to contain the temporal relations. Secondly, it describes very important minimisation operators – “contains” and “is-contained in” – that helps reduce temporal fault trees into their smallest forms. A procedure for analysing TBDDs is constructed and some examples of the use of TBDD are analysed.

Temporal Binary Decision Diagram Structure

TBDD is an extension of the traditional BDD approach. The most significant feature of TBDD is its ability to consider the temporal gates in its analysis. TBDD, just like the traditional BDD, is a directed acyclic graph with only two leaves: 0 encoding a success state and 1 encoding a failed state.

TBDD introduces a new structure called the If-Relation-Then-Else (*irte*) structure which is a slight extension of the traditional BDD If-Then-Else (*ite*) structure (Rauzy 1993). As an extension of BDDs *ite* structure, TBDD’s *irte* structure has an additional element to represent the temporal relation between an event and its left child event (conjunctive event). Fig. 4.2-1 is an expression and graphical representation of a basic TBDD *irte*.

$irte(X, O_x, f1, f2)$

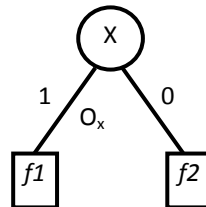


Figure 4.2-1: Basic TBDD *irte* structure

In the ordered quadruple node $irte(X, O_x, f1, f2)$, X is the Boolean variable or event represented by the node. O_x is the relationship between the current node and its conjunctive node on its 1 leaf. $f1$, and $f2$ are the logical functions on its 1-branch and 0-branch respectively. The *irte* is the If-Relation-Then-Else operation

```
if X with a relation  $O_x$  then  
    consider  $f1$   
else  
    consider  $f2$   
end if
```

O_x can be represented by '<', '|' or '&' for PAND, POR or SAND respectively. It can also be ':' for an end of a PAND operation or '-' for no temporal relation. If X is a basic event, the no temporal relation '-' and end of temporal relation ':' can be ignored.

Sub-tree Containment

Lozano and Valiente (2004) provide a technique for detecting if a tree is a sub tree of another tree. They represent binary trees with "well-formed parenthesis strings" they refer to as **balanced sequences**.

To obtain a balance sequence of a tree, the tree is traversed depth-first; a descent is represented with a '0' ascent with a '1'. For example, the ordered balanced sequence of the binary tree in Fig. 4.1-2 is 000010110110010111001011.

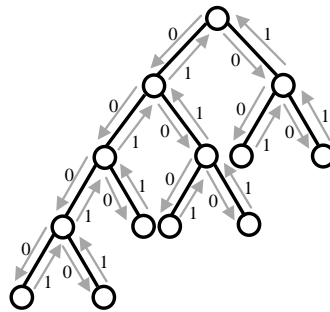


Figure 4.2-2: Directed binary tree

Due to the way they are structured and formed, balanced sequences will always have an even number of characters with equal amounts of '0's and '1's. They also always start from a '0' and end on a '1'. They do not take into account the node but rather the branches relating the nodes. Therefore, the names of the nodes do not matter.

A tree X is said to be contained in another Y if either $X = Y$ or there exist balanced sequences x_1, x_2, \dots, x_n and y_1, y_2, \dots, y_n with $x_i \subseteq y_i, 1 \leq i \leq n$, such that $X = x_1 x_2 \dots x_n$ and $Y = y_1 y_2 \dots y_n$. Therefore, the tree in Fig. 4.1-2 has the following sub-trees: 0101, 001011. Fig. 4.1-3 clearly shows how some of the sub-trees and how they are related.

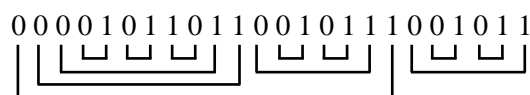


Figure 4.2-3: Balanced sequence

Lozano and Valiente 's (2004) technique cannot be applied to temporal fault trees for two major reasons:

1. Inclusion of nodes

From the balanced sequence, one cannot determine the types of nodes. Ignoring the node names, the balanced sequence of $A \bullet B$ is 0011 and the balanced sequence for $M \bullet N$ is 0011. Though $A \bullet B$ and $M \bullet N$ have the same balanced sequence, they are not necessarily equivalent. Adopting balanced sequences in reliability engineering requires that the nodes of binary tree from which balanced sequences will be generated include the events they represent. Meaning, the name of the node comes first in the balanced sequence and its branches are considered. The name of the node is written only once; it comes before the first descent branch and does not reappear after the last ascent branch.

Another issue to be resolved is the '0's and '1's used for descent and ascent respectively on branches and those used for 'success' and 'failed' states respectively. Using '0's and '1's for both situations will be chaotic and confusing. For the purpose of demonstration, the '0's used for descent and '1's used for ascent are represented by ' \downarrow ' and ' \uparrow ' respectively. If nodes/events are considered in balanced sequences and branches are represented with ' \downarrow ' and ' \uparrow ' for descent and ascent respectively, the balanced sequence of $A \bullet B$ in Fig. 4.1-4A will be $A \downarrow B \downarrow 1 \uparrow \downarrow 0 \uparrow \uparrow \downarrow 0 \uparrow$.

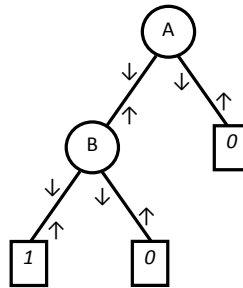


Figure 4.2-4: Directed binary tree of $A.B$

However, the inclusion of 'failed' and 'success' states makes analysis cumbersome since the balanced sequences are unnecessarily longer. They are therefore ignored in this thesis and the ' \downarrow ' and ' \uparrow ' are replaced with the original '0' and '1' respectively. $A \bullet B$ therefore, will have the balanced sequence A0B1. The 'failed' and 'success' states are disregarded.

2. Inclusion of Sequences

Traditionally, balance sequences do not consider the order in which nodes/events occur because they were not made to represent dynamic systems in the first place; they represent binary data structures.

Again, in this thesis, balanced sequences are modified slightly to include the dynamic relations existing between nodes. Every descent that is a temporal relation between a parent node and a child node is represented with the logical relational symbol '&', '<', '|', '.' or '+' instead of a '0'. Therefore, $A<B$ will have a balanced sequence $A<B1$.

For these reasons, the definitions of balanced sequence and sub-tree containments are redefined to include temporal behaviour. They are also renamed to **temporal balanced sequence** and **temporal sub-tree containment**.

Temporal balanced sequence:

*If X is a tree with n edges, the temporal balanced sequence of X , denoted by x , is a sequence over $\{0, *, 1\}$ (where '*' is either '<' or '&' or '|') of $3n + 1$ symbols defined as follows. The balanced sequence of a terminal node is an empty sequence. The balanced sequence of a non-terminal node is acquired by concatenating the balanced sequences of the children of that node, each of them preceded by the name of the node, a 0 for a descent and an additional 1 for an ascent. The balanced sequence of X is the balanced sequence of the root of X . A string x over $\{0, *, 1\}$ is a balanced sequence if there is a tree T such that t is its balanced sequence.*

Temporal Sub-tree Containment:

A tree X is said to be contained in another Y if either $X = Y$ or there exist balanced sequences x_1, x_2, \dots, x_n and y_1, y_2, \dots, y_n with $x_i \subseteq y_i$, $1 \leq i \leq n$ and all temporal symbols in Y changed to 0s, such that $X = x_1 x_2 \dots x_n$ and $Y = y_1 y_2 \dots y_n$.

Two important binary operators responsible for checking if a TBDD structure is contained in another are described as follows. The 'contains' operator, represented by ' \supseteq ', checks if a left hand operand contains or consists of a right hand operand. Conversely, the 'is contained in' operator, represented by ' \subseteq ', checks if a left hand operand is contained in or consists of a right hand operand. For example the balanced sequence B0C1, A0C1, B0D<E11, A0F1 and D<E1 will be contain in the balanced sequence A0B0C10D<E1110F1.

TBDD Procedure

To make TBDD analyses easier, all CSQs with parentheses have to be expanded before a ‘contains’ or ‘is contained in’ operation can take place. For example, using Pandora’s laws, $A < (B \cdot C)$ will be evaluated into $A < B \cdot C + A < C \cdot B$ before TBDD analysis can be undertaken. The order introduced by a temporal gate always overrides the order prescribed for a TBDD tree traversal. For example, if $A \rightarrow B$ is the order for a tree traversal, $B < A$ is $\text{irte}(B, <, \text{ite}(A, -, 1, 0), 0)$ instead of $\text{irte}(A, <, \text{ite}(B, -, 1, 0), 0)$.

TBDD structure is defined as follows. Given that:

$$A = \text{irte}(X, O_x, fx1, fx2)$$

$$B = \text{irte}(Y, O_y, fy1, fy2)$$

‘ $X \supseteq Y$ ’ = X contains Y and

‘ $X \subseteq Y$ ’ = X is contained in Y .

If $X \rightarrow Y$ (unless order in temporal term says otherwise), then:

$$\begin{aligned} A + B &= \text{irte}(X, O_x, fx1, fx2 + B) \\ &= \text{irte}(X, O_x, fx1, fx2 + \text{irte}(Y, O_y, fy1, fy2)) \end{aligned} \quad (4.2-1)$$

$$\begin{aligned} A \cdot B &= \text{irte}(X, O_x, fx1 \cdot B, fx2 \cdot B) \\ &= \text{irte}(X, O_x, fx1 \cdot \text{irte}(Y, O_y, fy1, fy2), fx2 \cdot \text{irte}(Y, O_y, fy1, fy2)) \end{aligned} \quad (4.2-2)$$

If $X=Y$ and $O_x = \text{'-'}$, then:

$$A + B = \text{irte}(X, O_x, fx1 + fy1, fx2 + fy2) \quad (4.2-3)$$

If $X=Y$ and $O_y = \text{'-'}$, then:

$$A \cdot B = \text{irte}(X, O_x, fx1 \cdot fy1, fx2 \cdot fy2) \quad (4.2-4)$$

If $X \subseteq Y$:

$$A + B = \text{irte}(X, O_x, fx1 + fy1, fx2 + fy2) \quad (4.2-5)$$

$$A \cdot B = \text{irte}(Y, O_y, fx1 \cdot fy1, fx2 \cdot fy2) \quad (4.2-6)$$

If $X \supseteq Y$:

$$A + B = \text{irte}(Y, O_y, fx1 + fy1, fx2 + fy2) \quad (4.2-7)$$

$$A \cdot B = \text{irte}(X, O_x, fx1 \cdot fy1, fx2 \cdot fy2) \quad (4.2-8)$$

If A is a set of CSQs and M is a subset of A such that for each member, i , of M , $M_i \subseteq B$:

$$A + B = (A - M_I) + (M_I + B) \quad (4.2-9)$$

If A is a set of CSQs and M is a subset of A such that for each member, i , of M , $B \subseteq M_i$:

$$A + B = (A - M) + (M + B) \quad (4.2-10)$$

Where ‘ $A-M$ ’ means A without the subset M .

PAND Gate Operations (Walker 2009)

$$1 \text{ PAND } B = B; 0 \text{ PAND } B = 0$$

The following are some examples of TBDD evaluations.

$$\begin{aligned} A + A < B &= irte(A, -, 1, 0) + irte(A, <, irte(B, :, 1, 0), 0) \\ &= irte(A, -, 1 + irte(B, -, 1, 0), 0 + 0) && \text{using (5)} \\ &= irte(A, -, 1, 0) = A \end{aligned}$$

$$\begin{aligned} A < B.A &= irte(A, <, irte(B, :, 1, 0), 0) \cdot irte(A, -, 1, 0) \\ &= irte(A, <, irte(B, -, 1, 0), 1, 0, 0) && \text{using (8)} \\ &= irte(A, <, irte(B, -, 1, 0), 0) = A < B \end{aligned}$$

$A < (B.C)$

In Pandora $a < (b.c) = a < b.c + a < c.b$, thus:

$$\begin{aligned} A < (B.C) &= irte(A, <, irte(B, :, irte(C, -, 1, 0), 0), 0) + irte(A, <, irte(C, :, irte(B, -, 1, 0), 0), 0) \\ &= irte(A, <, irte(B, :, irte(C, -, 1, 0), 0) + irte(C, :, irte(B, -, 1, 0), 0), 0) && \text{using (1)} \\ &= irte(A, <, irte(B, :, irte(C, -, 1, 0), irte(C, :, irte(B, -, 1, 0), 0), 0) \end{aligned}$$

$$\begin{aligned} A < (B.C) + A &= irte(A, <, irte(B, -, irte(C, :, 0, 1), irte(C, -, irte(B, -, 1, 0), 0), 0) + irte(A, -, 1, 0) \\ &= irte(A, -, 1, 0) = A && \text{using (7)} \end{aligned}$$

$$\begin{aligned} A < B < C + A < B &= irte(A, <, irte(B, <, irte(C, :, 1, 0), 0) + irte(A, <, irte(B, :, 1, 0), 0) \\ &= irte(A, <, irte(B, :, 1, 0), 0) && \text{using (7)} \end{aligned}$$

$$\begin{aligned} A < B < C + A < B + A &= irte(A, <, irte(B, :, 1, 0), 0) + irte(A, -, 1, 0) \\ &= irte(A, -, 1, 0) = A && \text{using (7)} \end{aligned}$$

For any CSQ without a temporal gate, the traditional BDD procedure is used in determining the resulting TBDD structure. However, if a temporal gate is present in a CSQ, then the new TBDD structure is employed.

Also, TBDD makes provision for contradictions. If a CSQ with a temporal gate appears on the ‘1’ branch of another CSQ with the same events but in reverse order, the entire branch is eliminated because it is a contradiction. For example the CSQ $A < B \cdot B < A$ or

$irte(A, <, irte(B, :, irte(B, <, irte(A, :, 1, 0), 0), 0), 0)$ will result in a contradiction and thus is eliminated.

4.2.2 Case Study: Hypothetical System

This section demonstrates the use of TBDD on a hypothetical system – Fig. 4.2-5. In the illustration, the top-event connects directly to nine (9) cut sequences. Two (2) of these cut sequences are critical events; four (4) have temporal gates and the remaining three (3) have only static gates. There are seven (7) events, five (5) temporal gates (PAND) and eight (8) static gates.

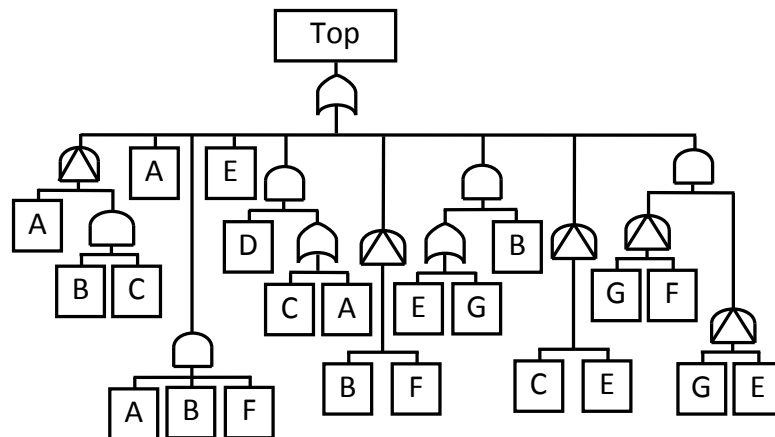


Figure 4.2-5: Hypothetical model

$$Top = A < (B.C) + A + A.B.F + E + D.(C+A) + B < F + (E+G).B + C < E + G < F.G < E$$

Following the TBDD structure, the first step in the analysis is to employ Boolean and temporal expansion laws.

$$A < (B.C) = A < B.C + A < C.B \quad \text{(using the Temporal Distributive Law)}$$

$$G < F.G < E = G < E < F + G < F < E$$

$$D.(C + A) = D.C + D.A \quad \text{(using the Boolean Distributive Law)}$$

$$(E + G).B = E.B + G.B$$

Therefore,

$$Top = A < B.C + A < C.B + A + A.B.F + E + D.C + D.A + B < F + E.B + G.B + C < E + G < E < F + G < F < E$$

Using the depth first traversal ($A \rightarrow E \rightarrow B \rightarrow F \rightarrow C \rightarrow D \rightarrow G$) for ordering the tree in 4.2-5:

$$A < B.C + A < C.B$$

$$= irte(A, <, irte(B, :, irte(C, -, 1, 0), 0), irte(A, <, irte(C, :, irte(B, -, 1, 0), 0), 0)) \quad \text{using (1)}$$

$$A < B.C + A < C.B + A$$

$$= irte(A, -, 1, 0) \quad \text{using (10, 4)}$$

$$A < B.C + A < C.B + A + A.B.F$$

$$= irte(A, -, 1, 0) \quad \text{using (5)}$$

$$A < B.C + A < C.B + A + A.B.F + E$$

$$= irte(A, -, 1, irte(E, -, 1, 0)) \quad \text{using (1)}$$

$$A < B.C + A < C.B + A + A.B.F + E + D.C$$

$$= irte(A, -, 1, irte(E, -, 1, irte(C, -, irte(D, -, 1, 0), 0))) \quad \text{using (5)}$$

$$A < B.C + A < C.B + A + A.B.F + E + D.C + D.A$$

$$= irte(A, -, 1, irte(E, -, 1, irte(C, -, irte(D, -, 1, 0), 0))) \quad \text{using (9, 5)}$$

$$A < B.C + A < C.B + A + A.B.F + E + D.C + D.A + B < F$$

$$= irte(A, -, 1, irte(E, -, 1, irte(C, -, irte(D, -, 1, 0), irte(B, <, irte(F, :, 1, 0), 0)))) \quad \text{using (1)}$$

$$A < B.C + A < C.B + A + A.B.F + E + D.C + D.A + B < F + E.B$$

$$= irte(A, -, 1, irte(E, -, 1, irte(C, -, irte(D, -, 1, 0), irte(B, <, irte(F, :, 1, 0), 0)))) \quad \text{using (9, 5)}$$

$$A < B.C + A < C.B + A + A.B.F + E + D.C + D.A + B < F + E.B + G.B$$

$$= irte(A, -, 1, irte(E, -, 1, irte(C, -, irte(D, -, 1, 0), irte(B, <, irte(F, :, 1, 0), irte(B, -, irte(G, -, 1, 0), 0)))))) \quad \text{using (1)}$$

$$A < B.C + A < C.B + A + A.B.F + E + D.C + D.A + B < F + E.B + G.B + C < E$$

$$= irte(A, -, 1, irte(E, -, 1, irte(C, -, irte(D, -, 1, 0), irte(B, <, irte(F, :, 1, 0), irte(B, -, irte(G, -, 1, 0), 0)))))) \quad \text{using (9, 5)}$$

$$A < B.C + A < C.B + A + A.B.F + E + D.C + D.A + B < F + E.B + G.B + C < E + G < E < F$$

$$= irte(A, -, 1, irte(E, -, 1, irte(C, -, irte(D, -, 1, 0), irte(B, <, irte(F, :, 1, 0), irte(B, -, irte(G, -, 1, 0), 0)))))) \quad \text{using (9, 5)}$$

$$Top = A < B.C + A < C.B + A + A.B.F + E + D.(C+A) + B < F + (E+G).B + C < E + G < E < F + G < F < E$$

$$= irte(A, -, 1, irte(E, -, 1, irte(C, -, irte(D, -, 1, 0), irte(B, <, irte(F, :, 1, 0), irte(B, -, irte(G, -, 1, 0), 0)))))) \quad \text{using (9, 5)}$$

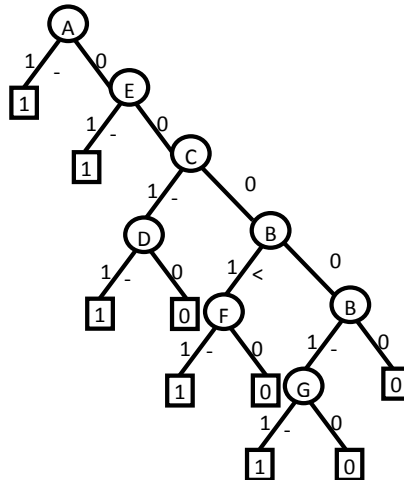


Figure 4.2-6: TBDD for $A \rightarrow E \rightarrow B \rightarrow F \rightarrow C \rightarrow D \rightarrow G$

Using the depth first traversal ($G \rightarrow D \rightarrow C \rightarrow F \rightarrow B \rightarrow E \rightarrow A$) for ordering the tree:

$$A < B.C = C.A < B$$

$$= \text{irte}(C, -, \text{irte}(A, <, \text{irte}(B, :, 1, 0), 0), 0)$$

$$A < C.B = B.A < C$$

$$= \text{irte}(B, -, \text{irte}(A, <, \text{irte}(C, :, 1, 0), 0), 0)$$

$$A < B.C + A < C.B$$

$$= \text{irte}(C, -, \text{irte}(A, <, \text{irte}(B, :, 1, 0), 0), 0) + \text{irte}(B, -, \text{irte}(A, <, \text{irte}(C, :, 1, 0), 0), 0)$$

$$= \text{irte}(C, -, \text{irte}(A, <, \text{irte}(B, :, 1, 0), 0), \text{irte}(B, -, \text{irte}(A, <, \text{irte}(C, :, 1, 0), 0), 0)) \quad \text{using (1)}$$

$$A < B.C + A < C.B + A$$

$$= \text{irte}(C, -, \text{irte}(A, <, \text{irte}(B, :, 1, 0), 0), \text{irte}(B, -, \text{irte}(A, <, \text{irte}(C, :, 1, 0), 0), 0)) + \text{irte}(A, -, 1, 0)$$

$$= (C.A < B + A) + (B.A < C + A) = A + A \quad \text{using (10)}$$

$$= \text{irte}(A, -, 1, 0) \quad \text{using (4)}$$

$$A < B.C + A < C.B + A + A.B.F$$

$$= \text{irte}(A, -, 1, 0) + \text{irte}(F, -, \text{irte}(B, -, \text{irte}(A, -, 1, 0), 0), 0)$$

$$= \text{irte}(A, -, 1, 0) \quad \text{using (5)}$$

$$A < B.C + A < C.B + A + A.B.F + E$$

$$= \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)) \quad \text{using (1)}$$

$$A < B.C + A < C.B + A + A.B.F + E + D.C$$

$$= \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)) + \text{irte}(D, -, \text{irte}(C, -, 1, 0), 0)$$

$$= (E + A) + D.C = E + A + D.C \quad \text{using (9)}$$

$$= \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0))) \quad \text{using (1)}$$

$$\begin{aligned}
& A < B.C + A < C.B + A + A.B.F + E + D.C + D.A \\
& = \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0))) + \text{irte}(D, -, \text{irte}(A, -, 1, 0), 0) \\
& = (D.C + E + A) + (D.A) = (D.C + E) + (A + D.A) = D.C + E + A \quad \text{using (9)} \\
& = \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)))
\end{aligned}$$

$$\begin{aligned}
& A < B.C + A < C.B + A + A.B.F + E + D.(C+A) + B < F \\
& = \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0))) + \\
& = \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(B, <, \text{irte}(F, :, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)))) \quad \text{using (1)}
\end{aligned}$$

$$\begin{aligned}
& A < B.C + A < C.B + A + A.B.F + E + D.(C+A) + B < F + B.E \\
& = \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(B, <, \text{irte}(F, :, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)))) + \text{irte}(B, -, \text{irte}(E, -, 1, 0), 0) \\
& = (D.C + E + A + B < F) + B.E = (D.C + A + B < F) + E + B.E \quad \text{using (9)} \\
& = D.C + A + E + B < F \\
& = \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(B, <, \text{irte}(F, :, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0))))
\end{aligned}$$

$$\begin{aligned}
& A < B.C + A < C.B + A + A.B.F + E + D.(C+A) + B < F + B.E + G.B \\
& = \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(B, <, \text{irte}(F, :, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)))) + \text{irte}(G, -, \text{irte}(B, -, 1, 0), 0) \\
& = \text{irte}(G, -, \text{irte}(B, -, 1, 0), \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(B, <, \text{irte}(F, :, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)))))) \quad \text{using (9)}
\end{aligned}$$

$$\begin{aligned}
& A < B.C + A < C.B + A + A.B.F + E + D.(C+A) + B < F + (E+G).B + C < E \\
& = \text{irte}(G, -, \text{irte}(B, -, 1, 0), \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(B, <, \text{irte}(F, :, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)))))) + \text{irte}(C, <, \text{irte}(E, :, 1, 0), 0) \\
& = (D.C + A + E + B < F + G.B) + C < E = (D.C + A + B < F + G.B) + (E + C < E) \\
& = D.C + A + E + B < F + G.B \quad \text{using (9)} \\
& = \text{irte}(G, -, \text{irte}(B, -, 1, 0), \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(B, <, \text{irte}(F, :, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0))))))
\end{aligned}$$

$$\begin{aligned}
& A < B.C + A < C.B + A + A.B.F + E + D.(C+A) + B < F + (E+G).B + C < E + G < E < F \\
& = \text{irte}(G, -, \text{irte}(B, -, 1, 0), \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(B, <, \text{irte}(F, :, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)))))) + \text{irte}(G, <, \text{irte}(E, <, \text{irte}(F, :, 1, 0), 0), 0) \\
& = (D.C + A + E + B < F + G.B) + G < E < F \\
& = (D.C + A + B < F + G.B) + (E + G < E < F) = D.C + A + E + B < F + G.B \quad \text{using (9)} \\
& = \text{irte}(G, -, \text{irte}(B, -, 1, 0), \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(B, <, \text{irte}(F, :, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0))))))
\end{aligned}$$

$$\begin{aligned} \text{Top} &= A < B.C + A < C.B + A + A.B.F + E + D.(C+A) + B < F + (E+G).B + C < E + \\ &G < E < F + G < F < E \end{aligned}$$

$$= \text{irte}(G, -, \text{irte}(B, -, 1, 0), \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(B, <, \text{irte}(F, :, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)))))) + \text{irte}(G, <, \text{irte}(F, <, \text{irte}(E, :, 1, 0), 0), 0)$$

$$= (D.C + A + E + B < F + G.B) + G < F < E$$

$$= (D.C + A + B < F + G.B) + E + G < F < E = (D.C + A + E + B < F + G.B) \quad \text{using (9)}$$

$$= \text{irte}(G, -, \text{irte}(B, -, 1, 0), \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(B, <, \text{irte}(F, :, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0))))))$$

$$= G.B + D.C + B < F + E + A$$

Using the depth first traversal (F→B→D→E→G→A→C) for ordering the tree:

$$A < B.C + A < C.B$$

$$= \text{irte}(A, <, \text{irte}(B, :, \text{irte}(C, -, 1, 0), 0), 0) + \text{irte}(B, -, \text{irte}(A, <, \text{irte}(C, :, 1, 0), 0), 0)$$

$$= \text{irte}(B, -, \text{irte}(A, <, \text{irte}(C, :, 1, 0), 0), \text{irte}(A, <, \text{irte}(B, :, \text{irte}(C, -, 1, 0), 0), 0)) \quad \text{using (1)}$$

$$A < B.C + A < C.B + A$$

$$= \text{irte}(B, -, \text{irte}(A, <, \text{irte}(C, :, 1, 0), 0), \text{irte}(A, <, \text{irte}(B, :, \text{irte}(C, -, 1, 0), 0), 0)) + \text{irte}(A, -, 1, 0)$$

$$= (B.A < C + A < B.C) + A = (B.A < C + A) + A < B.C = A + A < B.C = A \quad \text{using (10)}$$

$$= \text{irte}(A, -, 1, 0)$$

$$A < B.C + A < C.B + A + A.B.F$$

$$= \text{irte}(A, -, 1, 0) + \text{irte}(F, -, \text{irte}(B, -, \text{irte}(A, -, 1, 0), 0), 0)$$

$$= \text{irte}(A, -, 1, 0) \quad \text{using (5)}$$

$$A < B.C + A < C.B + A + A.B.F + E$$

$$= \text{irte}(A, -, 1, 0) + \text{irte}(E, -, 1, 0)$$

$$= \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)) \quad \text{using (1)}$$

$$A < (B.C) + A + A.B.F + E + D.C$$

$$= \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)) + \text{irte}(D, -, \text{irte}(C, -, 1, 0), 0)$$

$$= (E + A) + (D.C) = D.C + E + A \quad \text{using (9)}$$

$$= \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0))) \quad \text{using (1)}$$

$$A < (B.C) + A + A.B.F + E + D.C + D.A$$

$$= \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0))) + \text{irte}(D, -, \text{irte}(A, -, 1, 0), 0)$$

$$= (D.C + E + A) + D.A = (D.C + E) + A + D.A = D.C + E + A \quad \text{using (9)}$$

$$= \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)))$$

$$\begin{aligned}
& A < B.C + A < C.B + A + A.B.F + E + D.C + D.A + B < F \\
& = \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0))) + \text{irte}(B, <, \text{irte}(F, :, 1, 0), 0) \\
& = \text{irte}(B, <, \text{irte}(F, :, 1, 0), \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)))) \quad \text{using (1)}
\end{aligned}$$

$$\begin{aligned}
& A < B.C + A < C.B + A + A.B.F + E + D.C + D.A + B < F + E.B \\
& = \text{irte}(B, <, \text{irte}(F, :, 1, 0), \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)))) + \text{irte}(B, -, \text{irte}(E, -, 1, 0), 0) \\
& = (B < F + D.C + E + A) + B.E = (B < F + D.C + A) + B.E + E = B < F + D.C + E + A \quad \text{using (9)}
\end{aligned}$$

$$\begin{aligned}
& = \text{irte}(B, <, \text{irte}(F, :, 1, 0), \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)))) \\
& A < B.C + A < C.B + A + A.B.F + E + D.C + D.A + B < F + E.B + G.B \\
& = \text{irte}(B, <, \text{irte}(F, :, 1, 0), \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)))) + \text{irte}(B, -, \text{irte}(G, -, 1, 0), 0) \\
& = (B < F + D.C + E + A) + G.B = B < F + D.C + E + A + G.B \\
& = \text{irte}(B, <, \text{irte}(F, :, 1, 0), \text{irte}(B, -, \text{irte}(G, -, 1, 0), \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)))))) \quad \text{using (1)}
\end{aligned}$$

$$\begin{aligned}
& A < B.C + A < C.B + A + A.B.F + E + D.C + D.A + B < F + E.B + G.B + C < E \\
& = \text{irte}(B, <, \text{irte}(F, :, 1, 0), \text{irte}(B, -, \text{irte}(G, -, 1, 0), \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)))))) + \text{irte}(C, <, \text{irte}(E, :, 1, 0), 0) \\
& = (B < F + D.C + E + A + G.B) + C < E \\
& = (B < F + D.C + A + G.B) + E + C < E \\
& = B < F + D.C + E + A + G.B \\
& = \text{irte}(B, <, \text{irte}(F, :, 1, 0), \text{irte}(B, -, \text{irte}(G, -, 1, 0), \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)))))) \quad \text{using (9)}
\end{aligned}$$

$$\begin{aligned}
& A < B.C + A < C.B + A + A.B.F + E + D.C + D.A + B < F + E.B + G.B + C < E + G < E < F \\
& = \text{irte}(B, <, \text{irte}(F, :, 1, 0), \text{irte}(B, -, \text{irte}(G, -, 1, 0), \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)))))) + \text{irte}(G, <, \text{irte}(E, <, \text{irte}(F, :, 1, 0), 0), 0) \\
& = (B < F + D.C + E + A + G.B) + G < E < F \\
& = (B < F + D.C + A + G.B) + E + G < E < F \\
& = B < F + D.C + E + A + G.B \\
& = \text{irte}(B, <, \text{irte}(F, :, 1, 0), \text{irte}(B, -, \text{irte}(G, -, 1, 0), \text{irte}(D, -, \text{irte}(C, -, 1, 0), \text{irte}(E, -, 1, \text{irte}(A, -, 1, 0)))))) \quad \text{using (9)}
\end{aligned}$$

$$\begin{aligned}
Top &= A < B.C + A < C.B + A + A.B.F + E + D.C + D.A + B < F + E.B + G.B + C < E \\
&+ G < E < F + G < F < E \\
&= irte(B, <, irte(F, :, 1, 0), irte(B, -, irte(G, -, 1, 0), irte(D, -, irte(C, -, 1, 0), irte(E, -, 1, irte(A, -, 1, 0)))))) + irte(G, <, irte(E, <, irte(F, :, 1, 0), 0), 0) \\
&= (B < F + D.C + E + A + G.B) + G < F < E \\
&= (B < F + D.C + A + G.B) + E + G < F < E = B < F + D.C + E + A + G.B \quad \text{using (9)} \\
&= irte(B, <, irte(F, :, 1, 0), irte(B, -, irte(G, -, 1, 0), irte(D, -, irte(C, -, 1, 0), irte(E, -, 1, irte(A, -, 1, 0)))))) \\
&= B < F + B.G + D.C + E + A
\end{aligned}$$

It is evident that using all three traversal orders, $A \rightarrow E \rightarrow B \rightarrow F \rightarrow C \rightarrow D \rightarrow G$, $G \rightarrow D \rightarrow C \rightarrow F \rightarrow B \rightarrow E \rightarrow A$ and $F \rightarrow B \rightarrow D \rightarrow E \rightarrow G \rightarrow A \rightarrow C$ the results for the final minimized top event is the same: $A + E + B \cdot G + D \cdot C + B < F$. It is not guaranteed that this will always be the case; future work will be directed towards further investigation into this.

4.2.3 Strengths and Limitations TBDD Approach

Due to the techniques – ‘contains’ and ‘is-contained in’ – used in the minimization process, most ordering of events produce similar MCSQs. TBDD is not restricted to the size of a temporal fault tree; even though not proven yet, it is assumed that TBDD would be applicable on real world complex systems featuring only the static and PAND gates. The number of events in a temporal fault tree does not restrict its analytical capabilities.

Unfortunately, TBDD has some drawbacks. Unlike the traditional BDD analysis, TBDD is unable to produce the top-event probability from ‘unminimised’ CSQs. An area worth investigating is the development of a more efficient algorithm for the operations of the ‘contains’ and ‘is contained in’ operators. Furthermore, TBDD analyses can currently accommodate only the Priority-AND gate; thus there is also scope for future work in including other temporal gates.

Chapter Five

CASE STUDY

This chapter describes the functionality of an Aircraft Fuel System (AFS) and demonstrates how the qualitative and quantitative analysis of such a dynamic safety-critical system using Pandora can be achieved. It starts with the detailed description of AFS and continues to evaluate it qualitatively and then quantitatively. The novel GMMT approach described in chapter four is used in the qualitative analysis whilst the quantitative analysis (individual MCSQs and top-event probabilities) is evaluated with simulation and analytical techniques proposed in chapter three. The AFS used in this thesis is a modified version of that in Dheedan (2012).

5.1 Aircraft Fuel System (AFS)

Over the few past decades, aircraft have become one of the major means of international and intra-national transportation worldwide. There are increasing efforts to make them safer because their failure can lead to significant loss of life. This case study is an analysis of the fuelling system of an aircraft with two on-board engines and five fuel tanks. The AFS has two primary functions – storing and distributing fuel in the aircraft during both refuelling and consumption modes. Fuel is pumped via the fuelling point and is evenly distributed to all fuel storage tanks in the aircraft during refuelling mode. During the consumption mode however, the stored fuel is drawn to feed the aircraft's engine(s) at specific flow rates. Fig. 5.1-1 is a model of an AFS with port and starboard engines. AFS operates with the help of the following components:

- Five fuel storage tanks: these tanks are positioned along the horizontal axes of the aircraft to maintain a balance across its entire body. Two storage tanks are deployed to the port wing, two to the starboard wing and one in the centre of the fuselage. Fuel storage tanks are kept even and symmetrical along the horizontal axes to avoid any imbalances between the wings of the aircraft.
- Seven bi-directional pumps: individually embedded with speed sensors, these devices can pump fuel in a forward (positive) and reverse (negative) direction at variable rates to satisfy the port and starboard engines and maintain an even fuel level across the horizontal axes of the aircraft.

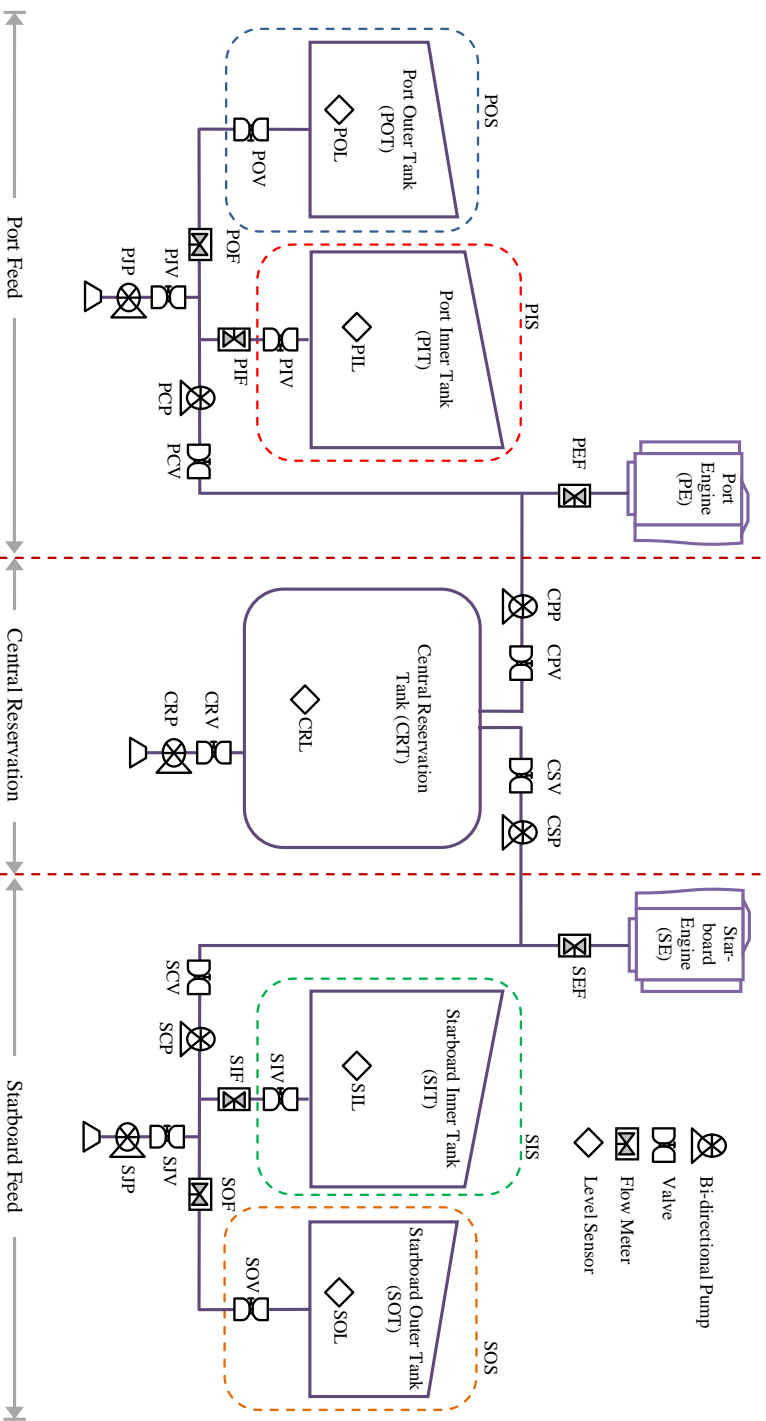


Figure 5.1-1: Aircraft Fuel System model

- CRV = Central Refuelling Valve
- CRP = central Refuelling Pump
- CRT = Central Reservation Tank
- CRL = Central Reservation Level Sensor
- CSP = Central Starboard Pump
- CSV = Central Starboard Valve
- SEF = Starboard Engine Flow meter
- SE = Starboard Engine
- SCV = Starboard Connecting Valve
- SCP = Starboard Connecting Pump
- SIF = Starboard Inner Flow meter
- SIV = Starboard Inner Valve
- SIT = Starboard Inner Tank
- SIL = Starboard Inner Level Sensor
- SOF = Starboard Outer Flow meter
- SOV = Starboard Outer Valve
- SOT = Starboard Outer Tank
- SOL = Starboard Outer Level Sensor
- SJV = Starboard Jettison Valve
- SJP = Starboard Jettison Pump
- PEF = Port Engine Flow meter
- PE = Port Engine
- PCV = Port Connecting Valve
- PCP = Port Connecting Pump
- PIF = Port Inner Flow meter
- PIV = Port Inner Valve
- PT = Port Inner Tank
- PIL = Port Inner Level Sensor
- POF = Port Outer Flow meter
- POV = Port Outer Valve
- POT = Port Outer Tank
- POL = Port Outer Level Sensor
- PJV = Port Jettison Valve
- PJP = Port Jettison Pump

- Thirteen valves: these devices allow fuel to flow in directions they requested by the pumps. They are shut to stop any flow of fuel through them. During the normal operation of AFS some valves are permanently shut whilst others are open to enable it achieve its dynamic functionality.
- Six Flow meters: each flow meter is responsible for measuring the volume of fuel flowing across it. Since the pumps are bi-directional, the flow meters are designed to determine the direction of flow also – negative or positive.
- Two jettison points: these are fuel outlets that have a valve and a pump each. The pumps draw and release fuel from associated tanks into the atmosphere via the valves during an in-flight emergency. This is done to make the aircraft lighter so that it can glide to a safer emergency landing.
- Fuel pipes: these are conduits that connect tanks to engines, refuelling point and jettison points and are interconnected with pumps, valves and flow meters.
- One Fuel Supply Control Unit (FSCU) : This is a centralised computer for controlling the entire AFS by providing the following functions:
 - o Ensure the even distribution of fuel to all five tanks during refuelling mode by controlling the valves and directions of pumps and determining the amount of fuel in each tank using the level sensors. This collaboration between the FSCU and the pumps, valves and level sensors ensure that all tanks are adequately fed and to appropriate levels to ensure an even balance of the aircraft.
 - o Ensure both port and starboard engines are evenly fed from appropriate tanks during consumption mode. This will also require the controlling of pumps, valves and flow metres to deliver the required amount of fuel to both engines to satisfy the demand thrust.
 - o Communicates the status (including measurements) of the AFS with other aircraft computing systems. This also involves receiving commands from the cockpit or other computing systems to control AFS's components and executing the command accordingly.

The AFS is divided into three sub-systems: Port Feed (PF) sub-system, Central Reservation (CR) sub-system and the Starboard Feed (SF) sub-system. PF and SF are further divided into various subsystems. PF has the Port Outer Sub-system (POS) and the Port Inner Sub-system (PIS); these are represented by the blue and red short-dashes lines respectively in Fig. 5.1-1. SF has the Starboard Outer Sub-system (SOS) and Starboard

Inner Sub-system (SIS); these are represented within brown and green short-dashes lines respectively in Fig. 5.1-1.

5.1.1 Operational Behaviour of AFS

The operations of AFS can be categorised in two main modes – refuelling and consumption – over the entire flight phase of the aircraft. The refuelling mode takes place before a flight (pre-flight) whilst the consumption mode occurs during a flight (in-flight), which includes the taxiing, take-off, climbing, cruising, approaching and landing phases. During the refuelling mode, the FSCU opens CRV to allow fuel flowing from the refuelling point into the CRT. CSP and CSV are activated to draw fuel from the CRT into SIT and SOT. Prior to this, SEV is shut to avoid fuel flowing into the engine. SJV is deactivated whilst SIV and SOV are regulated to allow flow into SIT and SOT respectively. The same operation takes place in the Port Feed system to refuel both inner and outer port tanks. After refuelling, the FSCU sets the valves and pumps in position, reverse to the refuelling mode, to supply fuel to both engines.

In normal operation, during consumption mode, PF tanks supply fuel to the port engine whilst the SF tanks feed the starboard engine. During this phase, the jettison valves are shut. The outer tanks are the primary sources of fuel to the engines while the inner tanks are the secondary sources. Therefore to run SE, a demand is placed on the FSCU which ensures that SE's 'thirst' is satisfied by the SOT first. In this case, SIV is closed and the SIS is dormant. If SOS reads 'empty' or SOF fails low (meaning, due to a fault, it reads a lower value than expected), SOV is shut and control is directed to draw fuel from SIT instead; in this case SOV is closed and SIF is activated. If SIS reads 'empty' or SIF fails low, SIV and SCV are closed to deactivate the SF sub-system and SE is fed from the CRT, which serves as a tertiary backup. The same order of operations applies to the PF sub-system.

To maintain a steady balance of the aircraft during the failure of any or all subsystems, FSCU communicates with another computing system, Aircraft Stability Control Unit (ASCU), which, with the use of other conduit/valve systems, ensures that fuel is evenly distributed across the horizontal axes of the plane. If for any reason, ASCU is not able to maintain a steady balance across the aircraft, it activates appropriate jettison valves and pumps to release fuel into the atmosphere to achieve a steady balance. A detailed description and functionality of ASCU is outside the scope of this thesis.

5.1.2 Temporal Fault Tree Analysis

The failure of SF, CR and PF is the failure of the entire fuelling system. This means there will be no fuel supply to the engines because the aircraft will be in a hazardous state – a state any pilot, crewmember or passenger will dread. If either SF or PF fails in addition to CR, the aircraft will be in a degraded state – only one engine will be operational. If anyone of the sub-systems – POS, PIS, SOS or SIS – fails, the aircraft remains in an operational state: both engines function normally. If any of the feed systems – PF or SF – fails, the aircraft remains operational because CR will substitute the failed feed system.

A typical top-event for AFS would be the failure of both starboard and port engines. However, since the aircraft is symmetrical, the qualitative analysis of SF is similar to that of PF. If the same components are used in building SF and PF then the quantitative analysis of both feed systems will be the same too. For this reason, we consider failure of the starboard feed (SF) system as the top-event used for constructing a temporal fault tree and qualitative and quantitative analysis of AFS.

To model the failure data of AFS, the following abbreviation scheme is adopted:

- ‘I’ is for internal failure of a components
- ‘C’ is for the inadvertent commission of a component
- ‘O’ is for omission of functionality of a component
- ‘Hi’ if fail high meaning the component reads an erroneous high value instead of an accurate lower value.

Failure of the starboard engines is due to omission of functionality from Starboard Engine Flow meter (SEF) (Fig. 5.1-2). Omission of SEF will occur:

- if Hi-SEF occurs strictly before an omission of functionality of SCV or
- if there is an omission in both SIS and SOS at exactly the same time or
- if there is an omission of SIS before an omission of SOS or
- if there is an omission of functionality of SCV and CSP.

$$O\text{-SEF} = \text{Hi-SEF} < O\text{-SCV} + O\text{-SOS} \& O\text{-SIS} + O\text{-SIS} < O\text{-SOS} + O\text{-SCV} . O\text{-CSP}$$

The starboard connecting valve (SCV) will fail:

- if it has an internal failure or
- if SOS fails and Hi-SEF occurs, both before SIS or both occurring without the occurrence of SIS or

- if there is an omission of functionality of SCP

$$O-SCV = I-SCV + (O-SOS \cdot Hi-SIF) | O-SIS + O-SCP$$

There will be no flow through the starboard connecting pump (SCP):

- if the pump is itself faulty or
- if there is an omission of functionality of both inner and outer starboard subsystems or
- if Hi-SOF occurs before the starboard outer subsystem fails:

$$O-SCP = I-SCP + O-SIS \cdot O-SOS + Hi-SOF < O-SOS$$

The Central Starboard Pump will fail:

- if there is an internal failure of CRL or
- if there is an internal failure of CSV or
- if CSP has an internal failure

$$O-CSP = I-CRL + I-CSV + I-CSP$$

The starboard outer subsystem will cease operation:

- if either SOV fails internally or
- SOL is faulty.

$$O-SOS = I-SOV + I-SOL$$

The starboard inner subsystem will also fail:

- if there is an internal failure of either SIV or
- SIL is faulty.

$$O-SIS = I-SIV + I-SIL$$

Figure 5.1-2 is a temporal fault tree for the failure of the starboard Feed (SF). The diagram is a graphical representation of the above failure descriptions.

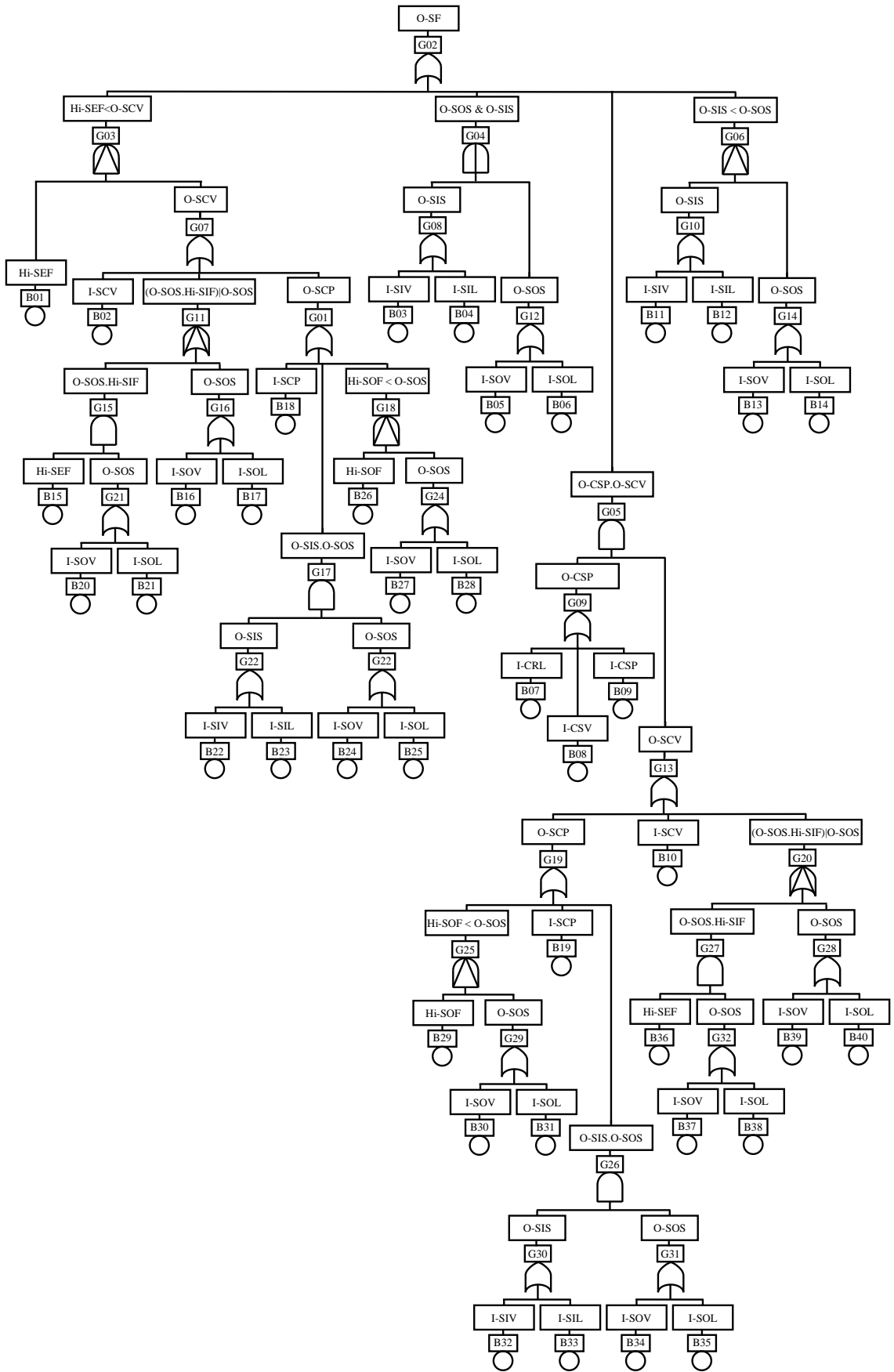


Figure 5.1-2: AFS temporal fault tree for “Failure of Starboard Feed”

5.2 Qualitative Analysis of AFS

As discussed earlier, Pandora can perform qualitative analysis using Euripides, Archimedes or a combination of both techniques. Unfortunately, Archimedes, due to its high demand for computing resources, is unable to analyse the AFS directly – the fault tree is too large. Euripides is unable to detect subtle completion within CSQs, and therefore the GMMT – combination of Archimedes and Euripides – is used. Below is an outline of the GMMT process on the AFS.

Classification 1:

1. Euripides

Evaluate O-SEF using Euripides produces one hundred and sixteen (116) CSQs listed in Appendix 3. Some of the CSQs contain doublets, which are decomposed further into many more CSQs. For example, the CSQ $[Hi-SEF/I-SOV].[Hi-SEF/I-SOL].[Hi-SEF/I-SCP].[Hi-SEF<I-SCV]$ produces eighty-five (85) CSQs having five distinct events each.

2. Cut-Off

To reduce the size of the CSQs in this case study, we implement a logical CSQ cut-off of 4. CSQs of the order five and above – meaning having more than five distinct events – are ignored in this analysis. We assume the combinations of events forming such cut sequences are practically impossible and therefore the probability of such a combination occurring is almost negligible. Applying the logical cut-off reduces the number of CSQs to 26.

$$\begin{aligned}
 & I-SIL.[I-SOV\&I-SIL].I-SIL.I-SOV.I-SOV + I-SOV.[I-SOV\&I-SIV]. \\
 & I-SOV.I-SIV.I-SIV + I-SOL.[I-SOL\&I-SIV].I-SOL.I-SIV.I-SIV + \\
 & [I-SIV<I-SOV].[I-SIV/I-SOL] + [I-SIV/I-SOV].[I-SIV<I-SOL] + \\
 & [I-SIL<I-SOV].[I-SIL/I-SOL] + Hi-SIF/I-SIL.Hi-SIF.I-CSP.I-SOV. \\
 & I-SOV + I-CRL.Hi-SIF.Hi-SIF.I-SOV.I-SOV + Hi-SIF.Hi-SIF.I-SOV. \\
 & I-SOV.I-CSV + Hi-SIF.Hi-SIF.I-CSP.I-SOL.I-SOL + Hi-SIF.Hi-SIF. \\
 & I-SOL.I-SOL.I-CSV + I-SCP.I-CSP+I-CRL.I-SCP+I-SCP.I-CSV + \\
 & I-CSP.I-SOL.I-SIL + I-CRL.I-SOL.I-SIL + I-SOL.I-SIL.I-CSV + \\
 & I-CSP.[Hi-SOF<I-SOV].[Hi-SOF/I-SOL] + I-CRL.[Hi-SOF<I-SOV]. \\
 & [Hi-SOF/I-SOL] + I-CSV.[Hi-SOF<I-SOV].[Hi-SOF/I-SOL] + I-CSP. \\
 & [Hi-SOF/I-SOV].[Hi-SOF<I-SOL] + I-CRL.[Hi-SOF/I-SOV].[Hi-SO< \\
 & I-SOL] + I-CSV.[Hi-SOF/I-SOV].[Hi-SOF<I-SOL] + I-CSP.I-SCV + \\
 & I-CRL.I-SCV + I-CSV.I-SCV
 \end{aligned}$$

As earlier mentioned, pSAND gates are not used in qualitative analysis. They are treated as ordinary SAND gates. However, once a temporal fault tree has been qualitatively analysed by Pandora to produce MCSQs, individual SAND gates can be examined to determine those that demonstrate the pSAND gate be-

haviour. When the pSAND gates have been determined, they are assigned their corresponding d values.

3. Grouping the above CSQs produces

	<i>Group 1</i>	<i>Group 2</i>
<i>Module 1:</i>	<i>I-CSP.I-SCV</i>	<i>I-CSP.I-SOL.I-SIL</i>
<i>Module 2:</i>	<i>I-CRL.I-SCV</i>	<i>I-CRL.I-SOL.I-SIL</i>
<i>Module 3:</i>	<i>I-CSV.I-SCV</i>	<i>I-SOL.I-SIL.I-CSV</i>
<i>Module 4:</i>	<i>I-SCP.I-CSP</i>	<i>[I-SIV<I-SOV].[I-SIV/I-SOL] +</i> <i>[I-SIV/I-SOV].[I-SIV<I-SOL] +</i> <i>[I-SIV<I-SOV].[I-SIV/I-SOL]</i>
<i>Module 5:</i>	<i>I-CRL.I-SCP</i>	<i>I-CRL.Hi-SIF.Hi-SIF.I-SOV.I-SOV</i>
<i>Module 6:</i>	<i>I-SCP.I-CSV</i>	<i>Hi-SIF.Hi-SIF.I-SOV.I-SOV.I-CSV</i>
<i>Module 7:</i>	<i>I-SIL.</i> <i>[I-SOV&I-SIL].</i> <i>I-SIL.I-SOV.I-SOV</i>	<i>Hi-SIF.Hi-SIF.I-CSP.I-SOL.I-SOL</i>
<i>Module 8:</i>	<i>I-SOV.</i> <i>[I-SOV&I-SIV].</i> <i>I-SOV.I-SIV. I-SIV</i>	<i>Hi-SIF.Hi-SIF.I-SOL.I-SOL.I-CSV</i>
<i>Module 9:</i>	<i>I-SOL.</i> <i>[I-SOL&I-SIV].</i> <i>I-SOL.I-SIV.I-SIV</i>	
	 <i>Group 3</i>	
<i>Module 1:</i>	<i>[Hi-SIF/I-SIL].Hi-SIF.I-CSP.I-SOV.I-SOV</i>	
<i>Module 2:</i>	<i>I-CSP.[Hi-SOF<I-SOV].[Hi-SOF/I-SOL] +</i> <i>I-CSP.[Hi-SOF/I-SOV].[Hi-SOF<I-SOL]</i>	
<i>Module 3:</i>	<i>I-CRL.[Hi-SOF<I-SOV].[Hi-SOF/I-SOL] +</i> <i>I-CRL.[Hi-SOF/I-SOV].[Hi-SOF<I-SOL]</i>	
<i>Module 4:</i>	<i>I-CSV.[Hi-SOF<I-SOV].[Hi-SOF/I-SOL] +</i> <i>I-CSV.[Hi-SOF/I-SOV].[Hi-SOF<I-SOL]</i>	

4. Regrouping step 3 produces the GMT below

	<i>Group 1</i>	<i>Group 2</i>
<i>Module 1:</i>	<i>I-CSP.I-SCV</i>	<i>I-CSP.I-SOL.I-SIL</i>
<i>Module 2:</i>	<i>I-CRL.I-SCV</i>	<i>I-CRL.I-SOL.I-SIL</i>
<i>Module 3:</i>	<i>I-CSV.I-SCV</i>	<i>I-SOL.I-SIL.I-CSV</i>
<i>Module 4:</i>	<i>I-SCP.I-CSP</i>	<i>I-SIV<I-SOV&I-SOL +</i> <i>I-SIV<I-SOV/I-SOL + I-SIV<I-SOL/I-SOV</i>
<i>Module 5:</i>	<i>I-CRL.I-SCP</i>	<i>I-CRL.Hi-SIF.I-SOV</i>
<i>Module 6:</i>	<i>I-SCP.I-CSV</i>	<i>Hi-SIF.I-SOV.I-CSV</i>
<i>Module 7:</i>	<i>I-SIL&I-SOV</i>	<i>Hi-SIF.I-CSP.I-SOL</i>
<i>Module 8:</i>	<i>I-SOV&I-SIV</i>	<i>Hi-SIF.I-SOL.I-CSV</i>
<i>Module 9:</i>	<i>I-SOL&I-SIV</i>	
	 <i>Group 3</i>	
<i>Module 1:</i>	<i>Hi-SIF<I-SIL.I-CSP.I-SOV + Hi-SIF<I-CSP/I-SIL. I-SOV+</i> <i>I-CSP<Hi-SIF/I-SIL.I-SOV + Hi-SIF&I-CSP/I-SIL.I-SOV +</i> <i>I-SOV<Hi-SIF/I-SIL.I-CSP + Hi-SIF<I-SOV/I-SIL.I-CSP +</i> <i>Hi-SIF&I-SOV/I-SIL.I-CSP</i>	

- Module 2:* $Hi-SOF<I-SOL<I-SOV.I-CSP + Hi-SOF<I-SOV<I-SOL.I-CSP + Hi-SOF<I-SOV&I-SOL.I-CSP + Hi-SOF<I-SOV|I-SOL.I-CSP + Hi-SOF<I-SOL|I-SOV.I-CSP$
- Module 3:* $Hi-SOF<I-SOL<I-SOV.I-CRL + Hi-SOF<I-SOV<I-SOL.I-CRL + Hi-SOF<I-SOV&I-SOL.I-CRL + Hi-SOF<I-SOV|I-SOL.I-CRL + Hi-SOF<I-SOL|I-SOV.I-CRL$
- Module 4:* $Hi-SOF<I-SOL<I-SOV.I-CSV + Hi-SOF<I-SOV<I-SOL.I-CSV + Hi-SOF<I-SOV&I-SOL.I-CSV + Hi-SOF<I-SOV|I-SOL.I-CSV + Hi-SOF<I-SOL|I-SOV.I-CSV$

Combination 1:

Only 3 modules can be combined. These are

- a) G1M7 and G3M1
- b) G1M8 and G2M4
- c) G1M9 and G2M4

The results of these combinations do not result in different CSQs therefore the GMT remains the same.

Comparison 1:

1. Declassification produces the following list of CSQs:

$I-CSP.I-SCV + I-CSP.I-SOL.I-SIL + I-CRL.I-SCV + I-CRL.I-SOL.I-SIL + I-CSV.I-SCV + I-SOL.I-SIL.I-CSV + I-SCP.I-CSP + I-SIV<I-SOV&I-SOL + I-SIV<I-SOV|I-SOL + I-SIV<I-SOL|I-SOV + I-CRL.I-SCP + I-CRL.Hi-SIF.I-SOV + I-SCP.I-CSV + Hi-SIF.I-SOV.I-CSV + I-SIL&I-SOV + Hi-SIF.I-CSP.I-SOL + I-SOV&I-SIV + Hi-SIF.I-SOL.I-CSV + I-SOL&I-SIV + Hi-SIF<I-SIL.I-CSP.I-SOV + Hi-SIF<I-CSP|I-SIL.I-SOV + I-CSP<Hi-SIF|I-SIL.I-SOV + Hi-SIF&I-CSP|I-SIL.I-SOV + I-SOV<Hi-SIF|I-SIL.I-CSP + Hi-SIF<I-SOV|I-SIL.I-CSP + Hi-SIF&I-SOV|I-SIL.I-CSP + Hi-SOF<I-SOL<I-SOV.I-CSP + Hi-SOF<I-SOV<I-SOL.I-CSP + Hi-SOF<I-SOV&I-SOL.I-CSP + Hi-SOF<I-SOV|I-SOL.I-CSP + Hi-SOF<I-SOL|I-SOV.I-CSP + Hi-SOF<I-SOL<I-SOV.I-CRL + Hi-SOF<I-SOV<I-SOL.I-CRL + Hi-SOF<I-SOV&I-SOL.I-CRL + Hi-SOF<I-SOV|I-SOL.I-CRL + Hi-SOF<I-SOL|I-SOV.I-CRL + Hi-SOF<I-SOL<I-SOV.I-CSV + Hi-SOF<I-SOV<I-SOL.I-CSV + Hi-SOF<I-SOV&I-SOL.I-CSV + Hi-SOF<I-SOV|I-SOL.I-CSV + Hi-SOF<I-SOL|I-SOV.I-CSV$

2. Classification: Euripides, Cut-off, Grouping produces

	<i>Group 1</i>	<i>Group 2</i>
<i>Module 1:</i>	$I-CSP.I-SCV$	$I-CSP.I-SOL.I-SIL$
<i>Module 2:</i>	$I-CRL.I-SCV$	$I-CRL.I-SOL.I-SIL$
<i>Module 3:</i>	$I-CSV.I-SCV$	$I-SOL.I-SIL.I-CSV$
<i>Module 4:</i>	$I-SCP.I-CSP$	$I-SIV<I-SOV&I-SOL + I-SIV<I-SOV I-SOL + I-SIV<I-SOL I-SOV$
<i>Module 5:</i>	$I-CRL.I-SCP$	$I-CRL.Hi-SIF.I-SOV$
<i>Module 6:</i>	$I-SCP.I-CSV$	$Hi-SIF.I-SOV.I-CSV$

<i>Module 7:</i>	<i>I-SIL&I-SOV</i>	<i>Hi-SIF.I-CSP.I-SOL</i>
<i>Module 8:</i>	<i>I-SOV&I-SIV</i>	<i>Hi-SIF.I-SOL.I-CSV</i>
<i>Module 9:</i>	<i>I-SOL&I-SIV</i>	<i>Hi-SOF<I-SOL.I-CSV</i>
<i>Module 10:</i>		<i>Hi-SOF<I-SOV.I-CSV</i>
<i>Module 11:</i>		<i>Hi-SOF<I-SOL.I-CRL</i>
<i>Module 12:</i>		<i>Hi-SOF<I-SOV.I-CRL</i>
<i>Module 13:</i>		<i>Hi-SOF<I-SOL.I-CSP</i>
<i>Module 14:</i>		<i>Hi-SOF<I-SOV.I-CSP</i>
<i>Module 15:</i>		<i>Hi-SIF&I-SOV.I-CSP +</i> <i>I-CSP<Hi-SIF.I-SOV +</i> <i>Hi-SIF<I-SOV.I-CSP +</i> <i>Hi-SIF<I-CSP.I-SOV +</i> <i>I-SOV<Hi-SIF.I-CSP +</i> <i>Hi-SIF&I-CSP.I-SOV</i>

Group 3

Module 1: Hi-SIF<I-SIL.I-CSP.I-SOV

Regrouping identifies a very subtle completion and produces

	<i>Group 1</i>	<i>Group 2</i>
<i>Module 1:</i>	<i>I-CSP.I-SCV</i>	<i>I-CSP.I-SOL.I-SIL</i>
<i>Module 2:</i>	<i>I-CRL.I-SCV</i>	<i>I-CRL.I-SOL.I-SIL</i>
<i>Module 3:</i>	<i>I-CSV.I-SCV</i>	<i>I-SOL.I-SIL.I-CSV</i>
<i>Module 4:</i>	<i>I-SCP.I-CSP</i>	<i>I-SIV<I-SOV&I-SOL +</i> <i>I-SIV<I-SOV/I-SOL +</i> <i>I-SIV<I-SOL/I-SOV</i>
<i>Module 5:</i>	<i>I-CRL.I-SCP</i>	<i>I-CRL.Hi-SIF.I-SOV</i>
<i>Module 6:</i>	<i>I-SCP.I-CSV</i>	<i>Hi-SIF.I-SOV.I-CSV</i>
<i>Module 7:</i>	<i>I-SIL&I-SOV</i>	<i>Hi-SIF.I-CSP.I-SOL</i>
<i>Module 8:</i>	<i>I-SOV&I-SIV</i>	<i>Hi-SIF.I-SOL.I-CSV</i>
<i>Module 9:</i>	<i>I-SOL&I-SIV</i>	<i>Hi-SOF<I-SOL.I-CSV</i>
<i>Module 10:</i>		<i>Hi-SOF<I-SOV.I-CSV</i>
<i>Module 11:</i>		<i>Hi-SOF<I-SOL.I-CRL</i>
<i>Module 12:</i>		<i>Hi-SOF<I-SOV.I-CRL</i>
<i>Module 13:</i>		<i>Hi-SOF<I-SOL.I-CSP</i>
<i>Module 14:</i>		<i>Hi-SOF<I-SOV.I-CSP</i>
<i>Module 15:</i>		<i>Hi-SIF.I-SOV.I-CSP</i>

Group 3

Module 1: Hi-SIF<I-SIL.I-CSP.I-SOV

3. Comparison: The final GMT in Comparison 1, Step 2 and Comparison 1, step 4 (because the combination process did not change the table) are different therefore, the entire Combination and Comparison is repeated.

Combination 2:

Again, only G1M7 and G3M1, G1M8 and G2M4, and G1M9 and G2M4 can be combined but they do not change the GMT.

Comparison 2:

Declassification and Classification produces the GMT below (note, only 2 groups):

	<i>Group 1</i>	<i>Group 2</i>
<i>Module 1:</i>	<i>I-CSP.I-SCV</i>	<i>I-CSP.I-SOL.I-SIL</i>
<i>Module 2:</i>	<i>I-CRL.I-SCV</i>	<i>I-CRL.I-SOL.I-SIL</i>
<i>Module 3:</i>	<i>I-CSV.I-SCV</i>	<i>I-SOL.I-SIL.I-CSV</i>
<i>Module 4:</i>	<i>I-SCP.I-CSP</i>	<i>I-SIV<I-SOV&I-SOL +</i> <i>I-SIV<I-SOV/I-SOL +</i> <i>I-SIV<I-SOL/I-SOV</i>
<i>Module 5:</i>	<i>I-CRL.I-SCP</i>	<i>I-CRL.Hi-SIF.I-SOV</i>
<i>Module 6:</i>	<i>I-SCP.I-CSV</i>	<i>Hi-SIF.I-SOV.I-CSV</i>
<i>Module 7:</i>	<i>I-SIL&I-SOV</i>	<i>Hi-SIF.I-CSP.I-SOL</i>
<i>Module 8:</i>	<i>I-SOV&I-SIV</i>	<i>Hi-SIF.I-SOL.I-CSV</i>
<i>Module 9:</i>	<i>I-SOL&I-SIV</i>	<i>Hi-SOF<I-SOL.I-CSV</i>
<i>Module 10:</i>		<i>Hi-SOF<I-SOV.I-CSV</i>
<i>Module 11:</i>		<i>Hi-SOF<I-SOL.I-CRL</i>
<i>Module 12:</i>		<i>Hi-SOF<I-SOV.I-CRL</i>
<i>Module 13:</i>		<i>Hi-SOF<I-SOL.I-CSP</i>
<i>Module 14:</i>		<i>Hi-SOF<I-SOV.I-CSP</i>
<i>Module 15:</i>		<i>Hi-SIF.I-SOV.I-CSP</i>

Combination 3:

Only G1M7 and G3M1, G1M8 and G2M4, and G1M9 and G2M4 can be combined but they do not change the GMT.

Comparison 3:

Declassification and classification produces the GMT in Comparison 3; this is the end of the qualitative process and the 26 CSQs in all the modules are the MCSQs. It is evident that there is no single point of failure.

5.3 Quantitative Analysis of AFS

Before any quantitative analysis (top-event probability and/or importance measures) of AFS is evaluated, basic component failure data are assigned to its components in Table 5.3-1.

These component failure data for various distributions are hypothetical data and not necessarily related; λ is the constant failure rate per hour for exponential distribution, α and β are the scale and shape parameters for Weibull distribution and μ_N and σ_N are the mean and standard deviation for a lognormal distribution.

Table 5.3-1: Component Failure Data

Component (<i>X</i>)	λ	α	β	μ_N	σ_N
I-SCP	5.84267E-5	3760	3.8	5.0235	1.1652
I-CSP	5.84267E-5	3760	3.8	5.0235	1.1652
I-SOV	1.65633E-3	535	0.7	7.0245	3.5152
I-SIV	1.65633E-3	535	0.7	7.0245	3.5152
I-CSV	1.65633E-3	535	0.7	7.0245	3.5152
I-SCV	1.65633E-3	535	0.7	7.0245	3.5152
I-CRL	2.21127E-6	4200	4.5	3.0125	1.1842
I-HiSOF	4.06861E-5	3510	3.8	6.0015	2.9332
I-HiSIF	4.06861E-5	3510	3.8	6.0015	2.9332
I-SOL	3.31774E-5	3490	3.2	8.0548	1.5122
I-SIL	3.31774E-5	3490	3.2	8.0548	1.5122

Table 5.3-2 contains various assumed *d* values for all pSAND gates. The durations are converted to hours during calculations.

Table 5.3-2: pSAND Gate Time Intervals

CSQ	interval (<i>d</i>) /seconds
I-SIL& _d I-SOV	0.1
I-SOV& _d I-SIV	0.2
I-SOL& _d I-SIV	0.3
I-SOV& _d I-SOL	0.4

MCSQs FV importance can be generated for all MCSQs achieved from the qualitative analysis of AFS and the component failure data tables above. Table 5.3-3 is a summary of MCSQ probabilities and importance measures for the first hour of operation using exponential distribution. It also has the top event probability for the same operational duration. In the table *MCSQ* is a minimal cut sequence, $Q(MCSQ)\{t\}$ is the unreliability of a MCSQ given a system lifetime *t*, $FV(MCSQ)\{t\}$ is the FV importance value for the MCSQ using equation (2.2-25) and *Rank* is the rank of the MCSQ with respect to its contribution to the occurrence of the top-event. Therefore, the first MCSQ in the table, *I-CSV.I-SCV*, contributes most to the top-event occurrence and *Hi-SOF<I-SOL.I-CRL* contributes least to the top-event occurrence. This demonstrates the usefulness of the

analytical techniques proposed in chapter three in the evaluation of MCSQs FV importance for dynamic safety-critical systems.

Table 5.3-3: MCSQs Probabilities and Importance for $t = 1$

MCSQ	Q(MSCQ){t}	FV(MCSQ){t}	RANK
I-CSV.I-SCV	2.7388894E-06	6.3113568E-01	1
I-SIV<I-SOV I-SOL	1.3694220E-06	3.1556260E-01	2
I-SCP.I-CSV	9.6690971E-08	2.2280973E-02	3
I-CSP.I-SCV	9.6690971E-08	2.2280973E-02	4
I-SIV<I-SOL I-SOV	2.7437861E-08	6.3226404E-03	5
I-CRL.I-SCV	3.6595572E-09	8.4328966E-04	6
I-SCP.I-CSP	3.4134798E-09	7.8658484E-04	7
I-SOV&I-SIV	3.0406895E-09	7.0068094E-04	8
I-CRL.I-SCP	1.2919329E-10	2.9770642E-05	9
Hi-SIF.I-SOV.I-CSV	1.1143246E-10	2.5677927E-05	10
I-SOL&I-SIV	9.1471872E-11	2.1078310E-05	11
I-SIL&I-SOV	3.0490631E-11	7.0261051E-06	12
Hi-SOF<I-SOV.I-CSV	8.9486793E-12	2.0620879E-06	13
Hi-SIF.I-SOV.I-CSP	3.9338985E-12	9.0650745E-07	14
Hi-SIF.I-SOL.I-CSV	2.2338789E-12	5.1476363E-07	15
I-SOL.I-SIL.I-CSV	1.8216190E-12	4.1976456E-07	16
Hi-SOF<I-SOV.I-CSP	3.1591509E-13	7.2797857E-08	17
Hi-SOF<I-SOL.I-CSV	1.7938591E-13	4.1336772E-08	18
I-CRL.Hi-SIF.I-SOV	1.4889008E-13	3.4309469E-08	19
I-SIV<I-SOV&I-SOL	1.0094928E-13	2.3262236E-08	20
Hi-SIF.I-CSP.I-SOL	7.8862591E-14	1.8172692E-08	21
I-CSP.I-SOL.I-SIL	6.4308586E-14	1.4818942E-08	22
Hi-SOF<I-SOV.I-CRL	1.1956746E-14	2.7552513E-09	23
Hi-SOF<I-SOL.I-CSP	6.3328580E-15	1.4593114E-09	24
I-CRL.I-SOL.I-SIL	2.4339496E-15	5.6086690E-10	25
Hi-SOF<I-SOL.I-CRL	2.3968584E-16	5.5231979E-11	26
Top Event	5.7562871E-05	NA	NA

Table 5.3-4 and 5.3-5 are the respective Fussell-Vesely (FV) and Birnbaum (BM) importances for Table 5.3-3. The former is computed using equation (2.2-25) whilst the latter is computed using (2.2-26). In Table 5.3-4, it can be seen that the biggest and least contributors to the top event occurrence are I-SCV and Hi-SOF respectively. Though I-SCP and I-CSP have the same failure data, they have different FV importance values because I-SCP is a primary pump serving SIT and SOT whereas I-CSP is a secondary pump serving CRT when I-SCP fails.

Table 5.3-5 lists the basic events in order of how the top event sensitivity to them – how a change in any affects the top event. Therefore a change in I-SCP has the biggest impact on the top event occurrence probability.

Table 5.3-4: FV Importances of Components $t=1hr$ (0.006 wks)

Component (X)	FV(X){t}	Rank
I-SCV	6.5420269E-01	1
I-CSV	6.5338818E-01	2
I-SOV	3.2259352E-01	3
I-SIV	3.2257882E-01	4
I-SOL	3.2187921E-01	5
I-SCP	2.3095310E-02	6
I-CSP	2.3066555E-02	7
I-CRL	8.7302168E-04	8
Hi-SIF	2.7149276E-05	9
I-SIL	7.4605964E-06	10
Hi-SOF	2.1803348E-06	11

Table 5.3-5: BM Importances of Components $t=1hr$ (0.006 wks)

Component (X)	BM(X){t}	Rank
I-SCV	1.0844213E-03	1
I-SCP	1.0844183E-03	2
I-CSV	1.0830726E-03	3
I-CSP	1.0830696E-03	4
I-CRL	1.0830687E-03	5
I-SIV	6.2180488E-04	6
I-SOV	4.3840699E-04	7

I-SOL	4.3804547E-04	8
Hi-SIF	1.8306665E-06	9
I-SIL	4.9542360E-07	10
Hi-SOF	1.8668020E-07	11

Tables 5.3-6, 5.3-7 and 5.3-8 are the top-event probabilities with various lifetimes for the exponential, Weibull and Lognormal distributions. ‘S-MC’ represents results for standard Monte Carlo simulation with 1×10^6 trials whilst ‘Exact’ represents the result from analytical techniques. Table 5.3-6 below contains results for the top event probability with various system lifetimes using the exponential distribution. The percentage errors are less than 10%. It is clear that for both simulation and analytical solutions, the system is due to fail after about 60 weeks of operation without repairs – component failures are non-repairable. System failure probability increases with time which is what is expected of every system with exponentially distributed components.

Table 5.3-6: AFS Top-Event Probabilities using Exponential Distribution

System Lifetimes	Top-Event Probabilities		
	Exact	S-MC	% Error
$t=1\text{hr}$ (0.006 wks)	4.3396207E-06	4.0000000E-06	7.8260E+00
$t=10\text{hr}$ (0.06 wks)	4.2746102E-04	4.5000000E-04	5.2728E+00
$t=100\text{hr}$ (0.6 wks)	3.6771307E-02	3.6429000E-02	9.3091E-01
$t=1000\text{hr}$ (6 wks)	7.9614228E-01	7.8337800E-01	1.6033E+00
$t=10000\text{hr}$ (60 wks)	9.9999999E-01	1.0000000E+00	7.4732E-07
$t=100000\text{hr}$ (600 wks)	1.0000000E+00	1.0000000E+00	0.0000E+00

The exact results have been calculated using the analytical solutions discussed in chapter three for Pandora’s gates. The simulation results have been estimated by modelling the system with the following simulation condition:

if ($r_{I-SCP} \leq F(I-SCP, t)$ && $r_{I-CSP} \leq F(I-CSP, t)$ // $r_{I-CRL} \leq F(I-CRL, t)$ && $r_{I-SCP} \leq F(I-SCP, t)$ // $r_{I-SCP} \leq F(I-SCP, t)$ && $r_{I-CSV} \leq F(I-CSV, t)$ // $r_{I-CSP} \leq F(I-CSP, t)$ && $r_{I-SCV} \leq F(I-SCV, t)$ // $r_{I-CRL} \leq F(I-CRL, t)$ && $r_{I-SCV} \leq F(I-SCV, t)$ // $r_{I-CSV} \leq F(I-CSV, t)$ && $r_{I-SCV} \leq F(I-SCV, t)$ // $r_{Hi-SIF} \leq F(Hi-SIF, t)$ && $r_{I-SOV} \leq F(I-SOV, t)$ && $r_{I-CSV} \leq F(I-CSV, t)$ // $r_{Hi-SIF} \leq F(Hi-SIF, t)$ && $r_{I-CSP} \leq F(I-CSP, t)$ && $r_{I-SOL} \leq F(I-SOL, t)$ // $r_{Hi-SIF} \leq F(Hi-SIF, t)$ && $r_{I-SOL} \leq F(I-SOL, t)$ && $r_{I-CSV} \leq F(I-CSV, t)$ // $r_{I-CSP} \leq F(I-CSP, t)$ && $r_{I-SOL} \leq F(I-SOL, t)$ && $r_{I-SIL} \leq F(I-SIL, t)$ // $r_{I-CRL} \leq F(I-CRL, t)$ && $r_{I-SOL} \leq F(I-SOL, t)$ && $r_{I-SIL} \leq F(I-SIL, t)$ //

```

r_I-SOL <= F(I-SOL, t) && r_I-SIL <= F(I-SIL, t) && r_I-CSV <= F(I-CSV, t) ||
r_I-CRL <= F(I-CRL, t) && r_Hi-SIF <= F(Hi-SIF, t) && r_I-SOV <= F(I-SOV, t) ||
r_Hi-SIF <= F(Hi-SIF, t) && r_I-SOV <= F(I-SOV, t) && r_I-CSP <= F(I-CSP, t) ||
((SimPAND(r_I-SIV, I-SIV, r_I-SOV, I-SOV, t) && r_I-SOL <= F(I-SOL, t) && TTF(I-
SOV, r_I-SOV, t) < TTF(I-SOL, r_I-SOL, t)))/(SimPAND(r_I-SIV, I-SIV, r_I-SOV,
I-SOV, t) && r_I-SOL > F(I-SOL, t))) || ((SimPAND(r_I-SIV, I-SIV, r_I-SOL, I-SOL, t)
&& r_I-SOV <= F(I-SOV, t) && TTF(I-SOL, r_I-SOL, t) < TTF(I-SOV, r_I-SOV, t)) ||
(SimPAND(r_I-SIV, I-SIV, r_I-SOL, I-SOL, t) && r_I-SOV > F(I-SOV, t))) ||
SimSAND(r_I-SIL, I-SIL, r_I-SOV, I-SOV, t, d1) || SimSAND(r_I-SOV, I-SOV, r_I-SIV,
I-SIV, t, d2) || SimSAND(r_I-SOL, I-SOL, r_I-SIV, I-SIV, t, d3) || r_I-SIV <= F(I-SIV, t)
&& SimSAND(r_I-SOV, I-SOV, r_I-SOL, I-SOL, t, d4) && TTF(I-SIV, r_I-SIV, t) <
TTF(I-SOV, r_I-SOV, t) && TTF(I-SIV, r_I-SIV, t) < TTF(I-SOL, r_I-SOL, t) ||
SimPAND(r_Hi-SOF, Hi-SOF, r_I-SOL, I-SOL, t) && r_I-CSV <= F(I-CSV, t) ||
SimPAND(r_Hi-SOF, Hi-SOF, r_I-SOV, I-SOV, t) && r_I-CSV <= F(I-CSV, t) ||
SimPAND(r_Hi-SOF, Hi-SOF, r_I-SOL, I-SOL, t) && r_I-CRL <= F(I-CRL, t) ||
SimPAND(r_Hi-SOF, Hi-SOF, r_I-SOV, I-SOV, t) && r_I-CRL <= F(I-CRL, t) ||
SimPAND(r_Hi-SOF, Hi-SOF, r_I-SOL, I-SOL, t) && r_I-CSP <= F(I-CSP, t) ||
SimPAND(r_Hi-SOF, Hi-SOF, r_I-SOV, I-SOV, t) && r_I-CSP <= F(I-CSP, t) )
end if

```

Where F, TTF, && and || retain their already defined meanings and SimPAND, SimSAND and SimPOR are algorithms for evaluating the PAND, SAND and POR conditions for the exponential distribution as described in chapter three. The CDF and its inverse are computed using the Numerics2.6.2 component from Math.NET (2013). Tables 5.3-7 and 5.3-8 are the top event probabilities for various lifetimes using the Weibull and Lognormal distributions respectively. There are no results for the analytical approach because techniques to quantitatively evaluate the PAND, POR and pSAND gates have not yet been developed for both distributions. From both results it is obvious that with increasing time, the probability of the top event occurring increases.

Table 5.3-7: AFS Top-Event Probabilities using Weibull Distribution

System Lifetimes	Top-Event Probabilities		
	Exact	S-MC	% Error
$t=1\text{hr}$ (0.006 wks)	NA	2.1100000E-04	NA
$t=10\text{hr}$ (0.06 wks)	NA	5.4200000E-03	NA
$t=100\text{hr}$ (0.6 wks)	NA	1.0338100E-01	NA
$t=1000\text{hr}$ (6 wks)	NA	7.4124600E-01	NA
$t=10000\text{hr}$ (60 wks)	NA	1.0000000E+00	NA
$t=100000\text{hr}$ (600 wks)	NA	1.0000000E+00	NA

Table 5.3-8: AFS Top-Event Probabilities using Lognormal Distribution

System Lifetimes	Top-Event Probabilities		
	Exact	S-MC	% Error
$t=1\text{hr}$ (0.006 wks)	NA	9.2400000E-04	NA
$t=10\text{hr}$ (0.06 wks)	NA	4.1609000E-02	NA
$t=100\text{hr}$ (0.6 wks)	NA	5.5378900E-01	NA
$t=1000\text{hr}$ (6 wks)	NA	9.8613400E-01	NA
$t=10000\text{hr}$ (60 wks)	NA	9.9999600E-01	NA
$t=100000\text{hr}$ (600 wks)	NA	1.0000000E+00	NA

‘Exact’ and ‘S-MC’ are the same as described earlier. Since both the Weibull and lognormal distributions have two parameters each, the same simulation condition is used to model both evaluations. The SimPAND, SimPOR and SimSAND for both simulations are evaluated with Algorithm 3.6-5, 3.6-6 and 3.6-7 respectively. However, the respective F and TTF functions are used for the distributions. The simulation condition for both the Weibull and lognormal simulation is:

```

if (  $r_{I-SCP} \leq F(I-SCP_a, I-SCP_b, t)$  &&  $r_{I-CSP} \leq F(I-CSP_a, I-CSP_b, t)$  ||
 $r_{I-CRL} \leq F(I-CRL_a, I-CRL_b, t)$  &&  $r_{I-SCP} \leq F(I-SCP_a, I-SCP_b, t)$  ||
 $r_{I-SCP} \leq F(I-SCP_a, I-SCP_b, t)$  &&  $r_{I-CSV} \leq F(I-CSV_a, I-CSV_b, t)$  ||
 $r_{I-CSP} \leq F(I-CSP_a, I-CSP_b, t)$  &&  $r_{I-SCV} \leq F(I-SCV_a, I-SCV_b, t)$  ||
 $r_{I-CRL} \leq F(I-CRL_a, I-CRL_b, t)$  &&  $r_{I-SCV} \leq F(I-SCV_a, I-SCV_b, t)$  ||
 $r_{I-CSV} \leq F(I-CSV_a, I-CSV_b, t)$  &&  $r_{I-SCV} \leq F(I-SCV_a, I-SCV_b, t)$  ||
 $r_{Hi-SIF} \leq F(Hi-SIF_a, Hi-SIF_b, t)$  &&  $r_{I-SOV} \leq F(I-SOV_a, I-SOV_b, t)$  &&
 $r_{I-CSV} \leq F(I-CSV_a, I-CSV_b, t)$  ||  $r_{Hi-SIF} \leq F(Hi-SIF_a, Hi-SIF_b, t)$  &&
 $r_{I-CSP} \leq F(I-CSP_a, I-CSP_b, t)$  &&  $r_{I-SOL} \leq F(I-SOL_a, I-SOL_b, t)$  ||
 $r_{Hi-SIF} \leq F(Hi-SIF_a, Hi-SIF_b, t)$  &&  $r_{I-SOL} \leq F(I-SOL_a, I-SOL_b, t)$  &&
 $r_{I-CSV} \leq F(I-CSV_a, I-CSV_b, t)$  ||  $r_{I-CSP} \leq F(I-CSP_a, I-CSP_b, t)$  &&  $r_{I-SOL} \leq F(I-SOL_a, I-SOL_b, t)$  &&
 $r_{I-SIL} \leq F(I-SIL_a, I-SIL_b, t)$  ||  $r_{I-CRL} \leq F(I-CRL_a, I-CRL_b, t)$  &&
 $r_{I-SOL} \leq F(I-SOL_a, I-SOL_b, t)$  &&  $r_{I-SIL} \leq F(I-SIL_a, I-SIL_b, t)$  ||
 $r_{I-SOL} \leq F(I-SOL_a, I-SOL_b, t)$  &&  $r_{I-SIL} \leq F(I-SIL_a, I-SIL_b, t)$  &&
 $r_{I-CSV} \leq F(I-CSV_a, I-CSV_b, t)$  ||  $r_{I-CRL} \leq F(I-CRL_a, I-CRL_b, t)$  &&
 $r_{Hi-SIF} \leq F(Hi-SIF_a, Hi-SIF_b, t)$  &&  $r_{I-SOV} \leq F(I-SOV_a, I-SOV_b, t)$  &&
 $r_{Hi-SIF} \leq F(Hi-SIF_a, Hi-SIF_b, t)$  &&  $r_{I-SOV} \leq F(I-SOV_a, I-SOV_b, t)$  &&
 $r_{I-CSP} \leq F(I-CSP_a, I-CSP_b, t)$  || (( $SimPAND(r_{I-SIV}, I-SIV_a, I-SIV_b, r_{I-SOV}, I-SOV_a, I-SOV_b, t)$  &&
 $r_{I-SOL} \leq F(I-SOL_a, I-SOL_b, t)$  &&  $TTF(I-SOV_a, I-SOV_b, r_{I-SOV}, t) < TTF(I-SOL_a, I-SOL_b, r_{I-SOL}, t)$  ||
( $SimPAND(r_{I-SIV}, I-SIV_a, I-SIV_b, r_{I-SOV}, I-SOV_a, I-SOV_b, t)$  &&  $r_{I-SOL} > F(I-SOL_a, I-SOL_b, t)$ 
))) || (( $SimPAND(r_{I-SIV}, I-SIV_a, I-SIV_b, r_{I-SOL}, I-SOL_a, I-SOL_b, t)$  &&
 $r_{I-SOV} \leq F(I-SOV_a, I-SOV_b, t)$  &&  $TTF(I-SOL_a, I-SOL_b, r_{I-SOL}, t) <$ 

```

```

TTF(I-SOV_a, I-SOV_b, r_I-SOV, t)) // (SimPAND(r_I-SIV, I-SIV_a, I-SIV_b, r_I-SOL,
I-SOL_a, I-SOL_b, t) && r_I-SOV > F(I-SOV_a, I-SOV_b, t)) // SimSAND(r_I-SIL,
I-SIL_a, I-SIL_b, r_I-SOV, I-SOV_a, I-SOV_b, t, d1) // SimSAND(r_I-SOV, I-SOV_a,
I-SOV_b, r_I-SIV, I-SIV_a, I-SIV_b, t, d2) // SimSAND(r_I-SOL, I-SOL_a, I-SOL_b,
r_I-SIV, I-SIV_a, I-SIV_b, t, d3) // r_I-SIV <= F(I-SIV_a, I-SIV_b, t) && SimSAND(r_I-
SOV, I-SOV_a, I-SOV_b, r_I-SOL, I-SOL_a, I-SOL_b, t, d4) && TTF(I-SIV_a, I-SIV_b,
r_I-SIV, t) < TTF(I-SOV_a, I-SOV_b, r_I-SOV, t) && TTF(I-SIV_a, I-SIV_b, r_I-SIV, t)
< TTF(I-SOL_a, I-SOL_b, r_I-SOL, t) // SimPAND(r_Hi-SOF, Hi-SOF_a, Hi-SOF_b,
r_I-SOL, I-SOL_a, I-SOL_b, t) && r_I-CSV <= F(I-CSV_a, I-CSV_b, t) //
SimPAND(r_Hi-SOF, Hi-SOF_a, Hi-SOF_b, r_I-SOV, I-SOV_a, I-SOV_b, t) &&
r_I-CSV <= F(I-CSV_a, I-CSV_b, t) // SimPAND(r_Hi-SOF, Hi-SOF_a, Hi-SOF_b,
r_I-SOL, I-SOL_a, I-SOL_b, t) && r_I-CRL <= F(I-CRL_a, I-CRL_b, t) //
SimPAND(r_Hi-SOF, Hi-SOF_a, Hi-SOF_b, r_I-SOV, I-SOV_a, I-SOV_b, t) &&
r_I-CRL <= F(I-CRL_a, I-CRL_b, t) // SimPAND(r_Hi-SOF, Hi-SOF_a, Hi-SOF_b,
r_I-SOL, I-SOL_a, I-SOL_b, t) && r_I-CSP <= F(I-CSP_a, I-CSP_b, t) //
SimPAND(r_Hi-SOF, Hi-SOF_a, Hi-SOF_b, r_I-SOV, I-SOV_a, I-SOV_b, t) &&
r_I-CSP <= F(I-CSP_a, I-CSP_b, t)
end if

```

From Table 5.3-6, the results of the analytical approach and Monte Carlo simulation estimation are similar. However, Tables 5.3-7 and 5.3-8 have no results for the analytical solution. A way of checking if the Monte Carlo simulation of a system converges with increasing trials achieved by simulating the model for various trials and plotting the values of the results against the trials. Figures 5.3-1 and 5.3-2 are charts for Monte Carlo estimates against various trials using the Weibull and lognormal distributions respectively.

The estimates have been calculated using a system lifetime, t , of 100 hours. From Figure 5.3-1 it can be seen that the results converge towards 0.1034 as seen in Table 5.3-7. Also in Figure 5.3-2 the estimates converge towards 0.5538 as seen in Table 5.3-8. A large number of trials have been used because no importance/dynamic sampling techniques have been used and the component failure parameters used in all experiments are relatively small.

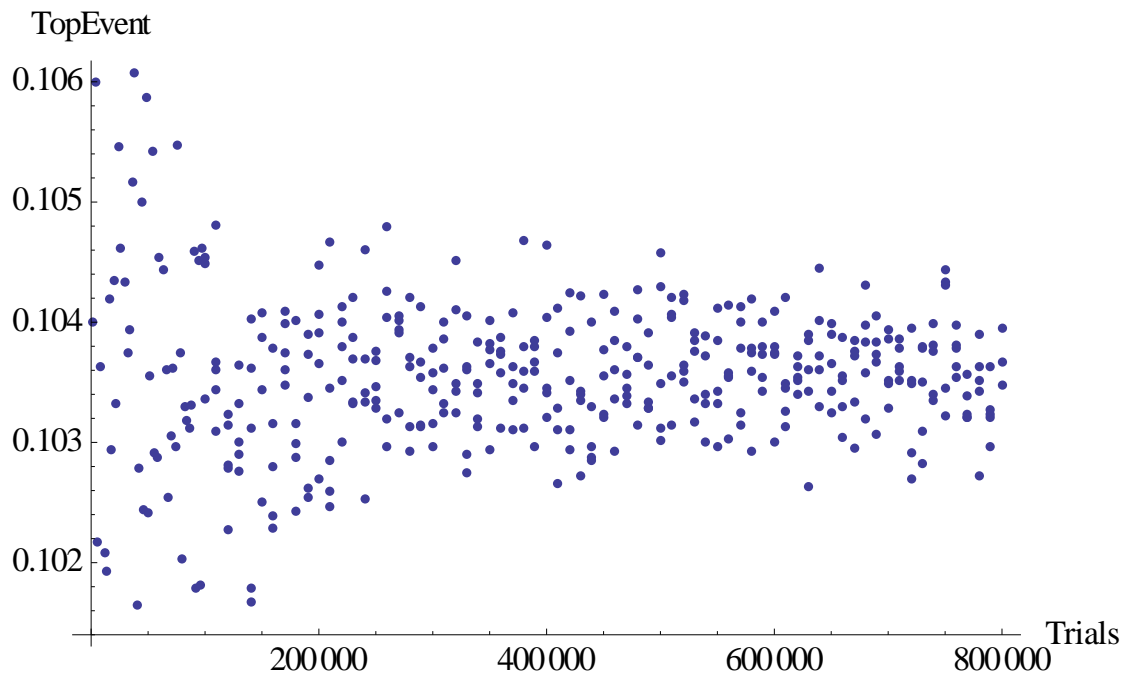


Figure 5.3-1: Monte Carlo estimates for Weibull distribution

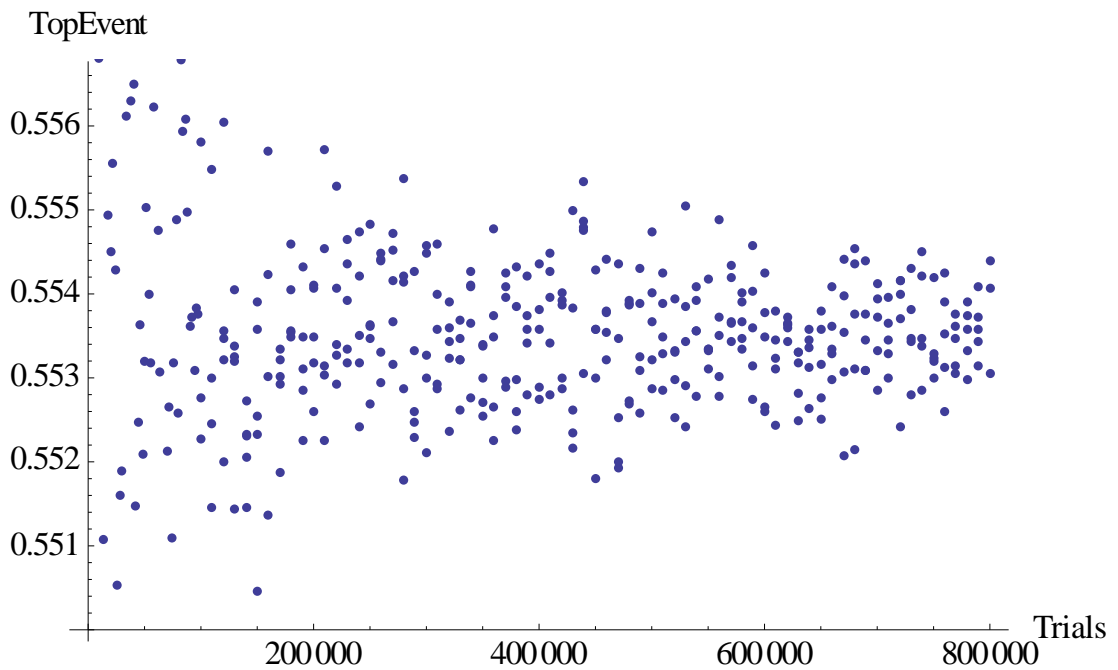


Figure 5.3-2: Monte Carlo estimates for Log-normal distribution

We use Mathematica (Wolfram 2011) to compute the best fit line and confidence intervals of both graphs using a linear regression. The results are summarized in Table 5.3-9. The lower and upper bounds are 95% confidence limits. The ‘best fit’ is the fitted model of the graph and ‘original’ is the initial Monte Carlo results achieved using $t = 100$ hours in Table 5.3-7 (for Weibull Distribution) and Table 5.3-8 (for Lognormal distribution).

Table 5.3-9: Characteristics of Figures 5.3-1 and Figure 5.3-2

Characteristics	Weibull Distribution	Lognormal Distribution
Lower Bound	0.103274	0.553400
Upper Bound	0.103545	0.553962
Best Fit	0.103409	0.553681
Original	0.103381	0.553789

Chapter Six

EVALUATION

This chapter evaluates the research described previously in this thesis and provides some suggestions for further research. Firstly, the techniques proposed in this thesis are compared against three of the techniques described in chapter two – Continuous Time Markov Chains, Bayesian Networks and Petri Nets. Secondly, the thesis contributions in chapters three and four are compared against the set research question and objectives in chapter one.

6.1 Evaluation of Techniques

In this subsection, two main evaluations are made: evaluation of quantitative techniques proposed in chapter three, and evaluation of qualitative techniques described in chapter four.

6.1.1 Evaluation of Quantitative Techniques

The main contribution of this thesis proposes two major quantitative techniques for evaluating temporal fault trees – analytical and simulation. In this subsection, both techniques are compared with each other and also compared against other techniques.

Analytical and Monte Carlo Simulation Techniques

The proposed analytical techniques in this thesis are algebraic expressions for the quantitative analysis of the POR, PAND and pSAND gates. These generic mathematical expressions have been formulated from first-principle. They can be applied to MCSQs with any amount of temporal gates. Analytical techniques are generally known to produce deterministic and exact solutions than simulation approaches. They are also generally more computationally efficient than simulation. Unfortunately, the analytical technique has some limitations: it is currently restricted to exponential distribution and may not be applicable to some complex systems as discussed in chapter two.

The simulation approach is evaluated using the Monte Carlo simulation; this is a concrete contribution of this thesis. It is popularly known that simulation is a powerful tool

for modelling and evaluating complex systems and is not restricted to any failure distribution. Wang and Pham (1997) report that Monte Carlo methods can be practically used to estimate the availability and unreliability of some complex systems that cannot be analytical analysed. However, unlike the analytical approach where a generic algorithm (like the RPN and Shunting-Yard) can be used in evaluating system failures, simulation models must be customized for every system; this can be time consuming and relatively difficult. In more detail, if $A < B|C + A \& D.E$ and $C|D.E + A < B \& G$ are the top-events of temporal fault trees X and Y , then generic analytical algorithm can be constructed to evaluate both X and Y , however, different simulation models must be constructed for these top-events. The generic Monte Carlo simulations proposed for the individual gate evaluations can be used for evaluating such expressions but it will be too computationally expensive to do so.

Evaluation of Proposed Techniques against CTMC

Markov chains are heavily used in the quantitative analysis of DFTs. Fault trees with dynamic behaviours can be modelled using Markov chains. Mathematical expressions can be extracted from these chains for quantitatively analysing a fault tree. In CTMC, Markov chains are used in the evaluation of CSQs with dynamic behaviours whilst BDD is used for the evaluation of static CSQs; this is done to harness the computational efficiency of BDDs.

Though continuous efforts are being made to improve Markov based techniques, they are known to be prone to the state-space explosion. This makes them error-prone, difficult and time consuming to construct and evaluate for large fault trees (Yevkin 2010). Unfortunately, they are also restricted to exponential distributions.

6.1.2 Evaluation of Qualitative Techniques

In this thesis two qualitative analyses techniques have been proposed – GMMT and TBDD. GMMT is a modularization technique whilst TBDD incorporates temporal behaviour into BDD. Evaluations of these proposed techniques are discussed below.

Groups and Modules Modularization Technique (GMMT)

The GMMT proposed in this thesis, is a simple modularization technique that categorizes CSQs into groups and CSQs with exactly the same type and number of events in CSQs into fields called modules. GMMT has been developed with the intention of par-

tially alleviating Pandora's inability to analyse large real-world temporal fault trees which is discussed in details in Walker (2009). Given a logical expression of a temporal fault tree, GMMT uses an algorithm which employs Archimedes and Euripides (described in chapter two) in generating the MCSQs.

GMMT can be used to evaluate dynamic safety-critical systems with several events as seen in the case study in chapter five. However, due to its use of Archimedes, it is restricted to CSQs with four or less events. Meaning, no single CSQs should have more than 4 distinct events though the temporal fault tree may have several hundreds of distinct events. Some reliability engineers may not consider this logical cut-off to be a limitation in some case studies; CSQs with more than four distinct events are less probable to occur.

One limitation of GMMT is its repeated use of Archimedes. This is not a direct limitation of GMMT as a modularization algorithm but due to its use of Archimedes. An ideal solution will be to have Archimedes evaluate the entire set of CSQs in one evaluation. However, at the moment, the Archimedes' algorithm is unable to analyse CSQs with more than 5 distinct events within a reasonable space of time (Walker 2009).

Temporal Binary Decision Diagrams (TBDDs)

A TBDD is a pioneering attempt to incorporate dynamic behaviours into Binary Decision Diagrams (BDDs) for quantitative and qualitative analysis. In this thesis, the TBDD approach proposed extends the if-then-else (ite) structure of a BDD (Rauzy 1993; Sinnamon 1996; Sinnamon & Andrews 1996) to include a temporal entity, r , representing the dynamic behaviour between two events. Thus TBDD uses a novel if-relation-then-else (irte) structure. A new procedure has been developed for qualitatively analysing TBDD structures. Apart from these two modifications, all other definitions and process in BDD are maintained in TBDD. At the moment, TBDD considers only the Priority-AND gate in its analysis.

The main advantage of BDD over its alternative approaches is its ability to provide quantitative information without the need of a prior qualitative analysis. Therefore, it has been proven to be more computationally efficient than the MCS or Monte Carlo approaches. This merit is the motivation for the development of TBDD. Unfortunately, TBDD is unable to exhibit this positive feature at the moment. TBDD uses a sub-tree containment algorithm to reduce CSQs into their smallest structures – minimization. For

this reason, it produces qualitative information from which temporal quantitative information can be computed. BDD is efficient but does not consider temporal information; TBDD considers some temporal information but is not as efficient as classical BDD.

6.2 Evaluation Against Objectives

The research question of this thesis states that:

How can the temporal fault tree of a safety-critical system featuring various failure distributions be quantitatively analysed using both simulation and analytical approaches?

From this statement, the research objectives stand out:

1. Probabilistic evaluation of temporal gates.
2. Probabilistic evaluation of MCSQs.
3. Probabilistic evaluation of top-event probability.

The probabilistic evaluation of a system can be broken down into various units as shown in Fig. 6.1-1:

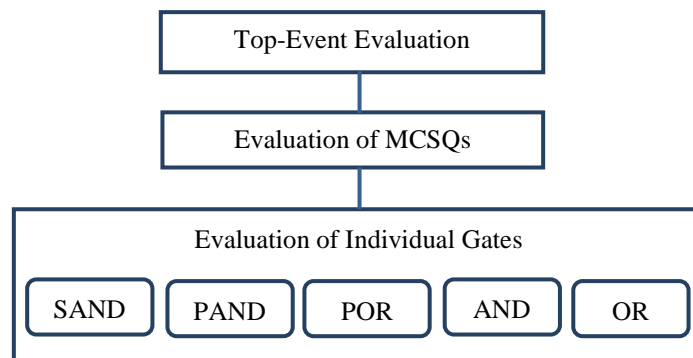


Figure 6.1-1: Breakdown of thesis' aim

Therefore, the quantitative analysis (top-event probability and importance measures) of a total system failure is dependent on the quantitative analysis of individual MCSQs, which is also dependent on the quantitative analysis of individual logic gates; this is the structure by which the objectives are evaluated.

Evaluation of Top-Event Probability

As already mentioned, apart from classical BDD analysis, the top-event failure probability and importance measures of a dynamic system can be calculated when the probability of MCSQs have been determined. At a top level, a system failure can be expressed in terms of a Disjunctive Normal Form (DNF); this is a disjunction of MCSQs. This means the top level operator of a system failure is an OR gate. This DNF representation of system failure is not new; it has been adopted in most PSA techniques since the inception of FTA.

As stated in chapter 2, there are two popular analytical techniques for evaluating the OR gate: the Esary-Proschan (Eqn. 2.2-22) and PIE (Eqn. 2.2-23). For a fault tree with n MCSQs, PIE will require $\sum_{k=0}^n k \times \binom{n}{k} - 1$ arithmetic calculations to evaluate a system failure whilst Esary-Proschan will require only $2 \times n$ arithmetic calculations. Both techniques provide very similar results; this makes EP more computationally efficient than PIE. Apart from the analytical approaches, a simulation alternative using Monte Carlo can be used. This involves the construction of a Monte Carlo model of individual MCSQs and ‘ORing’ them to produce the system failure probability. Unfortunately, Monte Carlo simulation cannot be used to evaluate importance sampling as ‘easily’ as its analytical counterparts; this will be discussed in detail later.

Evaluation of MCSQs

In classical fault tree analysis, quantitative analysis of minimal cut sets (smallest combinations of events that can cause a system failure) are done by evaluating the conjunction (AND) gates followed by the disjunction (OR) gates. In temporal fault tree analysis, a similar approach is adopted; however, due to the presence of temporal gates an extended operator precedence is prescribed. Using the analytical approach, this precedence order is simple to implement and evaluate. However, this can be more technically challenging when simulation is used. For example, to quantitatively evaluate $A \prec B | C$ analytically, one would only need to evaluate $(A \prec B) | C$. The evaluation of $A \prec B | C$ with simulation using the order of precedence can be done but it is too computationally expensive. This is because two simulations would be required: a large number of trials would need to be executed to evaluate $A \prec B$ and another large number of trials would be required to simulate $(A \prec B) | C$.

A more efficient simulation approach is to have one simulation model for $A < B | C$ without literally using the precedence order. In such an approach, the time-to-failure (TTF) of components is used to model temporal behaviours in MCSQs. Since a MCSQ may contain two or more temporal behaviours, there may be the need for more TTF comparisons in the simulation model; this can be problematic. Thankfully, the solution to this challenge is to construct the TTF comparison on a logical sequence in the MCSQ and not on the precedence order. More details and examples of the construction of MCSQs models for simulation have been provided in section five of chapter three.

Evaluation of the POR Gate

The usage and importance of the Priority-OR (POR) gate have been discussed in Edifor *et al.* (2012) – where the POR gate is used to model some specific situations which difficult and cumbersome to model with ordinary fault trees. Apart from the fact that it simplifies a temporal fault tree with the POR behaviour, it has a set of laws for its qualitative analysis (Walker 2009). An earlier effort (Merle 2010) to quantify the POR gate provided an algebraic solution for evaluating a POR gate with only two events.

In this thesis, a generic mathematical expression has been formulated for the quantitative analysis of MCSQs with two or more POR gates. This has been done from three different expressions: logical definition in Pandora, Markov chains and some of Pandora's laws. Also a Monte Carlo simulation model has been constructed for estimation of MCSQs with the same or different combinations of temporal and static gates.

Evaluation of the PAND

The Priority-AND gate represents the situation where an event occurs before another event. There are generally two types of PAND gates: Inclusive PAND (iPAND) where an event occurs before or at the same time as another event or exclusive PAND (ePAND) where an event occurs strictly before another event. Pandora uses the ePAND in its qualitative analysis. It has been proven in chapter three that these gates provide different qualitative interpretations but quantitatively both gates evaluate to zero. Various techniques for quantitatively analysing iPAND analytically have been identified and adopted for ePAND quantification. Unfortunately, these analytical techniques are restricted to exponential distribution. This thesis has proposed a simulation alternative – using Monte Carlo – which is not restricted to a particular distribution and can be used for the analysis of MCSQs with any number of ePAND gates.

Evaluation of the SAND Gate

The SAND gate models situations where two or more events occur at exactly the same time. A mathematical expression has been formulated in this thesis from all three completion laws in Pandora's to quantitatively model the SAND gate. Interestingly, this expression evaluates to zero. This means that it is technically nearly impossible for two or more exponentially distributed mutually exclusive events to occur at exactly the same time. This result is not new and has been widely discussed (Merle, Roussel, Lesage & Bobbio 2010). Therefore, though the SAND gate is significant in qualitative analysis it cannot be used for quantitative analysis. For this reason, no attempts have been made to evaluate the SAND gate using simulation or to evaluate it in other failure distributions.

Evaluation of the pSAND Gate

There are situations where the occurrence of two events within a relatively short duration of time can trigger the occurrence of an output event; these scenarios are known as 'nearly simultaneous' scenarios. Although there are various descriptions of this scenario, in this thesis, a peculiar type of the nearly simultaneous events – parameterised Simultaneous-AND (pSAND) – has been described and quantitatively analysed. pSAND is represented by $\&_d$ where, d is the duration of interval between the occurrence of the input events. The main difference between the pSAND gate and other nearly simultaneous descriptions are:

1. The pSAND gate is commutative; $A \&_d B = B \&_d A$
2. The pSAND gate does not occur if the duration between its input events is greater than d .
3. pSAND is mainly used in quantitative analyses and not a formal description of the nearly simultaneous scenario.

An analytical model and quantitative analysis of the pSAND gate with multiple input events have been developed in this thesis. A Monte Carlo simulation for the pSAND gate has also been developed in this thesis from a simulation model. Though the analytical solution is restricted to exponential distribution, the simulation model can be applied in other distributions. Unfortunately, the pSAND gate is currently not used in qualitative analysis; details of this limitation are discussed below.

6.3 Limitations and Further Research

Though this thesis has made some significant contributions, there remain a number of areas for potential future development: the techniques proposed and developed in the thesis could be improved in terms of their functionality. Another development would be to apply the techniques in other fields apart from reliability engineering. Below are some limitations that can lead to further research

Repairable Events

One of the assumptions made in this thesis is that all systems under consideration are non-repairable. Meaning, when a component fails, it remains in the failed state and is not repaired. Therefore for all system lifetime durations defined in this thesis, a component's operational state can only move from a success state to a failed state and not vice-versa.

However, in the real world, many safety-critical systems experience wears, malfunctions and the likes, which may require repairs of components. Some fault tree techniques such as the DFT and classical FTA have quantitative techniques for analysing repairable systems (Kumamoto *et al.* 1980; Boudali *et al.* 2007). It will be a worthwhile effort to develop techniques for quantitatively analysing temporal fault trees of systems which are repairable. This will improve the practicability and scalability of Pandora on real-world case studies.

pSAND Logical Analysis

The pSAND gate is a novel gate that represents the occurrence of two or more events within a relatively short duration of time. After qualitative analysis of temporal fault trees, MCSQs with SAND gates are examined for the pSAND behaviour. All MCSQs with SAND gates but that do not exhibit the pSAND behaviour are ignored from quantitative analysis whilst the remaining MCSQs with SAND gates are quantified using the pSAND formula. The inclusion of the pSAND gate into qualitative analysis of temporal fault trees is a potential field for future research. One way of including the pSAND gate will be the redefinition of the PAND and POR gates to include specific time. For example, a new parameterized-PAND (pPAND) and parameterized-POR (pPOR) could be defined to complement the pSAND gate in such a way that:

$$A \cdot B = (A \text{ pPAND } B) + (A \text{ pSAND } B) + (B \text{ pPAND } A)$$

Where A and B will occur when A occurs before B within a duration or A and B occur within a duration or B occurs before A within a duration. It is worth noting that this is only a proposed solution that will require further investigation. It is not clearly known if the existing laws in Pandora will accommodate pPAND, pSAND and pPOR; perhaps novel laws must be formulated.

Temporal Binary Decision Diagram (TBDD)

TBDD is an initial attempt to quantify temporal fault trees with BDDs. TBDD includes only the PAND gate in its analysis; it is unable to consider the SAND or POR gates in its analysis. Some real world dynamic safety-critical systems feature more behaviours than the AND, OR and PAND behaviours. For TBDD to be more applicable to real world case studies, it must be extended to consider other temporal gates.

Monte Carlo Simulation

As discussed earlier, the standard Monte Carlo technique is very effective for modelling complex systems. However it requires relatively higher computing resources to simulate models before estimating their failure probabilities in reliability engineering. This problem is not new and various efforts have been made to improve this. Some of these techniques include importance sampling which concentrates on acquiring random number for simulation from a favourable domain of values.

Though analytical solutions are generally more computationally efficient than Monte Carlo simulations in solving a problem, it has been shown, in this thesis, that Monte Carlo simulation can be used for problems where analytical solutions have not yet been developed. Unfortunately, the Monte Carlo simulation used in this thesis has no importance sampling technique or dynamic stopping technique to improve its computational efficiency.

However, towards the completion of this thesis, the author has identified a recent and efficient dynamic stopping technique (Meedeniya *et al.* 2011). Though this has not been used in this thesis, a glimpse of its effects can be seen in Table 6.3-1 and Table 6.3-2; the latter is an extension of the former. The results in both tables have been calculated using the exponential distribution of AFS case study (Table 5.3-6) described in chapter five.

Table 6.3-1: Effect of dynamic stopping technique 1

Mission Time	Exact Solution	S-MC APPX	DS-MC APPX
1E+01	4.2746102E-04	4.1500000E-04	3.753955E-04
1E+02	3.6771307E-02	3.6571000E-02	3.762376E-02
1E+03	7.9614228E-01	7.8379300E-01	7.776000E-01
1E+04	9.9999999E-01	1.0000000E+00	1.000000E+00
1E+05	1.0000000E+00	1.0000000E+00	1.000000E+00

In Table 6.3-1, “S-MC” is the result for standard Monte Carlo simulation and “DS-MC” is the result for Monte Carlo simulation with the dynamic stopping technique. In table 6.3-2, “S-MC DEV” and “DS-DEV” are the deviations for the standard and dynamic stopping Monte Carlo techniques whilst “S-MC TIME” and “DS-MC TIME” are the execution times (in seconds) for both techniques.

Table 6.3-2: Effect of dynamic stopping technique 2

Mission Time	S-MC DEV	DS-MC DEV	S-MC TIME	DS-MC TIME
1E+01	2.9151E+00	1.2180E+01	3.2884217	1.4949691
1E+02	5.4474E-01	2.3183E+00	4.9749460	1.5529996
1E+03	1.5511E+00	2.3290E+00	5.8943707	1.5820028
1E+04	7.4732E-07	7.4732E-07	6.2754881	1.5945726
1E+05	0.0000E+00	0.0000E+00	6.5254955	1.6035880

In Table 6.3-2, it is evident that though the dynamic sampling technique is generally more deviated than the standard technique, its execution time is considerably lower than that of the standard simulation technique – approximately four times faster.

Another limitation associated with the use of Monte Carlo simulation is its inability to calculate importance measures. For example, to calculate the FV importance of particu-

lar MCSQs, one will find the ratio of the MCSQ to the top-event probability. As already discussed, the most appropriate way of evaluating the top-event probability of a system using Monte Carlo is to model the entire system and evaluate it. Unfortunately, using this method does not permit the evaluation of importance measures at the moment. A crude method will be to model, simulate and evaluate MCSQ individually before using them for importance analysis. However, this method requires more computational resources and is prone to deviation from the accurate results due to multiple rounding-ups. Potential future efforts can be directed towards an efficient Monte Carlo model that will allow the calculation of both top-event probabilities and importance measures.

Application in Other Fields

In this thesis, Pandora has been applied to the automotive, electrical and aircraft industries. Future research can be focused on using Pandora to evaluate other fields of study outside the scope covered in this thesis; a plausible field is Supply Chain Risk Management (SCRM). So far, classical FTA is used in evaluating the reliability of some supply chains (Vanany *et al.* 2009). To make Pandora applicable in SCRM:

- a. the PAND gate could represent a situation where the occurrence of an output event (failure) is caused by the occurrence of an event followed by another event. For example, assuming a company gets its supplies (raw materials) for production by two means of transportation: trucks (primary) and trains (secondary). It also has a supplies department that is responsible for ordering supplies via the trains system when the truck system fails. There will be no supplies to the company if the supplies department fails to order for more supplies (perhaps due to employee strike or unavailability) before the train system fails.
- b. the pSAND gate could be used to represent the situation where the occurrence of two failures within a duration of time could trigger the occurrence of another bigger failure. For example, if two deliveries to a manufacturing plant do not arrive (fail delivery targets) at the plant within a specific duration of time for manufacturing to take place, the manufacturing of a batch will be delayed leading to financial losses.
- c. the POR gate could be used to represent situations where the occurrence of an event before another or the occurrence of the first event alone can lead to the occurrence of a supply chain failure.

An example of the application of Pandora in SCRM is described in Appendix 4.

Incorporation into HiP-HOPS

HiP-HOPS (Hierarchically Performed Hazard Origin and Propagation Studies) is a technique for the semi-automatic construction and evaluation of fault trees FMEAs (Papadopoulos *et al.* 2001). HiP-HOPS operates by annotating the model of a system with expressions representing its component failure behaviour, examining the model and finally constructing and evaluating fault trees and FMEAs generated from the model. Due to HiP-HOPS' semi-automated ability to construct and evaluate fault trees and FMEAs, it offers system designers the opportunity of performing FTA and FMEA more easily than most conventional ways which are not automated.

Unfortunately, HiP-HOPS is restricted to static gates. Future research efforts could be channelled into:

1. investigating how Pandora could be incorporated into HiP-HOPS using GMMT.
2. quantitatively evaluating HiP-HOPS' resulting temporal fault trees using the quantitative techniques discussed in this thesis.

Chapter Seven

CONCLUSION

State-of-the-art safety-critical systems promise numerous benefits. However, they introduce new hazards and risks. It is the priority of dependability engineers to ensure that such risks and hazards are minimised as much as reasonably possible to improve such systems' reliability, availability, maintainability, safety and integrity. Fault Tree Analysis (FTA) is one of the earliest techniques used in reliability engineering to perform off-line evaluation of safety-critical systems but, unfortunately, is unable to capture the sequential/dynamic failures of components; this leads to the inaccurate qualitative/quantitative evaluation of system reliabilities.

Temporal Fault Tree analysis – via Pandora – is one of the recent improvements made to solve this problem by the use of the combinatorial AND and OR gates and novel temporal gates: exclusive Priority-AND (ePAND), Simultaneous-AND (SAND), Priority-OR (POR).

Pandora's strength lies in its approach to qualitative analysis; it provides several laws for a comprehensive logical analysis of temporal fault trees of dynamic safety-critical systems. However, it is unable to provide any quantitative analysis, which provides measures of how reliable/unreliable a system is and how relative contributions, sensitivities or importances of component failures lead to a total system failure. This leads to the research question of this thesis:

How can the temporal fault tree of a safety-critical system featuring various failure distributions be quantitatively analysed using both simulation and analytical approaches?

Techniques to improve the quantitative analyses of temporal fault trees should be able to evaluate probabilistically ePAND, SAND and POR and combinations of them. Also, they should be able to evaluate the total system failure probability and relative component importances analytically or with simulation.

In this thesis, analytical expressions have been formulated and simulations developed to satisfy the aims and objectives stated in chapter one; these are the contributions of this thesis are described below.

1. Probabilistically evaluate individual temporal logic gates:

ePAND: Various solutions (from algebraic modelling, Markov chain analysis, first principle, Petri nets, Bayesian nets and Poisson process) for evaluating iPAND have been identified. It has been mathematically proven in this thesis that though iPAND and ePAND probabilistically evaluate to zero, they mean different things logically. The evaluation of ePAND has therefore been deduced from a mathematical formulation of the iPAND.

SAND: It has been mathematically proven in this thesis, from some of Pandora's logical laws, that for any statistically independent events the SAND gate equates to zero.

POR: In this thesis, a mathematical expression has been formulated for quantitatively evaluating the POR gate; this has been done from first principle (calculus), Markov Chains and Pandora's laws.

pSAND: There are situations where the occurrence of two or more events within a relatively short duration of time can lead to the occurrence of another event. Such situations are known as nearly simultaneous events. In this thesis, the description of such nearly simultaneous situations – parameterised Simultaneous-AND (pSAND) – is made. pSAND, unlike Common Cause Failure (CCF), considers an interval of duration, d , within which its input events must occur to trigger the occurrence of an output event. pSAND is also different from many nearly simultaneous scenarios described in literature because it is commutative: A pSAND B is equivalent to B pSAND A . Probabilistic evaluation of the pSAND gate has been deduced in this thesis and applied to a real world safety-critical system.

2. Probabilistically evaluate Minimal Cut Sequences (MCSQs):

Combinations of component failures with static gate have different behaviour from combinations of component failures temporal gates. Not only has techniques been developed for evaluating combinations of components failures with different temporal and static gates, simulation conditions necessary for such expressions with both temporal and static gates to be evaluated with Monte Carlo have been developed. Therefore, the quantitative analysis of MCSQs can be achieved both with analytical and simulation approaches.

3. Probabilistically evaluate the top event and importance measures:

The evaluation of individual gates and combinations of such these gates make the evaluation of total system failure probability possible. Analytical and simulation approaches for evaluating the top-event probability have been developed. In the analytical approach, as in most analytical approaches, the top-event is calculated after individual MCSQs are evaluated whilst in the simulation approach, the entire system is modelled using conditions for each MCSQs occurring and simulated a large number of times to estimate the top-event probability. The evaluation of top-event probability using the analytical approach makes the evaluation of individual component contribution to the top-event occurrence possible. However, with simulation, only the top-event can be estimated; importance measures cannot be estimated at the moment.

4. Monte Carlo Simulation of total system failure for various distributions:

All analytical solutions for quantitatively analysing temporal fault trees made in this thesis are restricted to exponential distribution. Monte Carlo simulations of temporal fault trees of system failures with other distributions, other than the exponential distribution, have been developed. This has been achieved by using the time-to-failures of components with temporal behaviours and constructing a simulation condition to satisfy such situations. Once these conditions for various MCSQs are developed, a simulation model of the entire system can be constructed and simulated. Once these have been done, the failure of the system can be estimated. Apart from exponential distribution, these techniques have been used for analysing systems with Weibull and Lognormal distributions. Unfortunately, these simulation approaches are unable to evaluate the relative component importances contributing to the system failure.

5. Improvement to logical analysis technique:

Quantitative analysis of temporal fault trees using either analytical or simulation approaches in this thesis cannot be done without prior qualitative analysis. However, the qualitative approaches used in evaluating temporal fault trees have some limitations. In this thesis, two techniques have been described for improving the qualitative analysis of temporal fault trees. The first technique is a modularization technique known as Groups and Modules Modularization Technique (GMMT) which arranges fault trees in terms of groups and modules and evaluates these repeatedly using Archimedes and Euripides – logical techniques for

analysing temporal fault trees – until minimal cut sequences are achieved. The second technique, known as Temporal Binary Decision Diagram (TBDD), involves the inclusion of temporal behaviour into Binary Decision Diagrams. In addition to the traditional AND and OR gates, GMMT considers all three temporal gates – PAND, SAND and POR – whilst TBDD considers only the PAND gate.

Currently, the novel pSAND gate is not considered in logical analysis – only in probabilistic analysis. The inclusion of pSAND into the logical analysis of Pandora is a limitation that requires future efforts. A possible improvement is the introduction of novel gates – parameterised PAND and parameterised POR instead of PAND and POR – into the logical analysis to complement it. In this thesis, all Monte Carlo simulations are standard simulations where a system is modelled and simulated – without any dynamic stopping technique. However, system failures are relatively small and require simulations of large numbers of trials for more accurate evaluations.

Another area requiring improvement is the implementation of a dynamic stopping technique in Monte Carlo simulation which will improve its computational efficiency in terms of computing resources and time. Finally, it is worth noting that many real world systems are repairable, where a component fails and does not remain in a failed state because it can be repaired. This, unfortunately, contrasts one of the assumptions in this thesis; all component failures are non-repairable. One particular area of future work would be to develop novel approaches or improve the existing approaches to evaluate repairable system.

Reliability engineering is a sensitive and critical area engineers, system designers, stakeholders and system users do not take for granted because of the devastating effects systems can have if they should fail. This thesis discusses some of the improvements made in reliability engineering and proposes techniques for quantitatively analysing dynamic safety-critical systems using both analytical and simulation approaches. These techniques are useful for engineers who wish to quantitatively analyse complex state-of-the-art systems to find out how reliable they really are.

REFERENCES

- Akers, S.B., 1978. Binary Decision Diagrams. *IEEE Transactions on Computers*, C-27(6), pp.509–516. Available at:
<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1675141>.
- Aldemir, T., Miller, D.W., Stovsky, M.P., Kirschenbaum, J., Bucci, P., Fentiman, A.W. & Mangan, L.T., 2006. Current State of Reliability Modeling Methodologies for Digital Systems and Their Acceptance Criteria for Nuclear Power Plant Assessments (NUREG/CR-6901), Washington DC: US Nuclear Regulatory Commission. Available at: <http://pbadupws.nrc.gov/docs/ML0608/ML060800179.pdf>.
- Alverbro, K., Nevhage, B. & Erdeniz, R., 2010. *Methods for Risk Analysis*, Stockholm. Available at: https://www.etde.org/etdeweb/details_open.jsp?osti_id=979763 [Accessed August 14, 2013].
- Amari, S., Dill, G. & Howald, E., 2003. A New Approach to Solve Dynamic Fault Trees. In *Reliability and Maintainability Symposium (RAMS)*. pp. 374–379. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1182018 [Accessed October 18, 2013].
- Andrews, J.D., 2000. To NOT or Not To NOT. *Proceedings of the 18th International System Safety Conference*, Fort Worth, (September).
- BBC, 2009a. 1986: Soviets Admit Nuclear Accident. *On This Day*. Available at: http://news.bbc.co.uk/onthisday/hi/dates/stories/april/28/newsid_2500000/2500975.stm [Accessed August 5, 2009].
- BBC, 2009b. Seven Dead In Space Shuttle Disaster. *On This Day*. Available at: http://news.bbc.co.uk/onthisday/hi/dates/stories/january/28/newsid_2506000/2506161.stm [Accessed August 5, 2013].
- Bedford, T. & Cooke, R., 2001. *Probabilistic Risk Analysis: Foundations and Methods*, Cambridge: Cambridge University Press.
- Bobbio, A. & Franceschinis, G., 2003. Parametric fault tree for the dependability analysis of redundant systems and its high-level petri net semantics. ... , *IEEE Transactions on*, 29(3), pp.270–287. Available at:
<http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1183940> [Accessed November 6, 2012].
- Bobbio, A., Portinale, L., Minichino, M. & Ciancamerla, E., 2001. Improving the Analysis of Dependable Systems by Mapping Fault Trees into Bayesian Networks. *Re-*

- liability Engineering & System Safety, 71(3), pp.249–260. Available at:
<http://linkinghub.elsevier.com/retrieve/pii/S0951832000000776>.
- Boole, G., 1916. *Collected Logical Works* 2nd ed., London: The Open Court Publishing Company.
- Van der Borst, M. & Schoonakker, H., 2001. An Overview of PSA Importance Measures. *Reliability Engineering & System Safety*, 72(3), pp.241–245. Available at: <http://linkinghub.elsevier.com/retrieve/pii/S0951832001000072>.
- Boudali, H., Crouzen, P. & Stoelinga, M., 2007. Dynamic Fault Tree Analysis Using Input/Output Interactive Markov Chains. In *IEEE/IFIP International Dependable Systems and Networks*. Available at:
http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4273022 [Accessed October 8, 2013].
- Boudali, H. & Dugan, J.B., 2006. A Continuous-Time Bayesian Network Reliability Modeling and Analysis Framework. *IEEE Transactions on Reliability*, 55(1), pp.86–97.
- Boudali, H. & Dugan, J.B., 2005. A Discrete-Time Bayesian Network Reliability Modeling and Analysis Framework. *Reliability Engineering & System Safety*, 87(3), pp.337–349. Available at:
<http://linkinghub.elsevier.com/retrieve/pii/S0951832004001310> [Accessed October 9, 2013].
- BP, 2010. BP Releases Report on Causes of Gulf of Mexico Tragedy. Generic Article. Available at:
<http://www.bp.com/genericarticle.do?categoryId=9034548&contentId=7064928> [Accessed August 5, 2013].
- Bremaud, P., 1999. *Markov Chains: GIBBS Fields, Monte Carlo Simulation, and Queue*, New York: Springer-Verlag.
- Bryant, R., 1986. Graph-Based Algorithms for Boolean Function Manipulation. *Computers*, *IEEE Transactions on*, C-35(8), pp.677 – 691. Available at:
http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1676819 [Accessed August 20, 2013].
- BSI, 1979. *Glossary of Terms Used In Quality Assurance (Including Reliability and Maintainability Terms)*, London: British Standards Institution.
- Buck, S., 1997. *Applying Probabilistic Risk Assessment to Agricultural Nonpoint Source Pollution*. Virginia Polytechnic Institute and State University. Available at:

- <http://scholar.lib.vt.edu/theses/public/etd-17391653976940/etd.pdf> [Accessed June 15, 2014].
- Burks, A., Warren, D. & Wright, J., 1954. An Analysis of a Logical Machine Using Parenthesis-Free Notation. *Mathematical tables and other aids to ...*, 8(46), pp.53–57. Available at: <http://www.jstor.org/stable/10.2307/2001990> [Accessed July 30, 2013].
- Chaochen, Z., Hoare, C.A.R. & Ravn, A.P., 1991. A Calculus of Durations.pdf. *Information Processing Letters*, 40(5), pp.269–276. Available at: <http://www.sciencedirect.com/science/article/pii/002001909190122X>.
- Chiacchio, F., Compagno, L., D’Urso, D., Manno, G. & Trapani, N., 2011. Dynamic Fault Trees Resolution: A Conscious Trade-Off Between Analytical and Simulative Approaches. *Reliability Engineering & System Safety*, 96(11), pp.1515–1526. Available at: <http://linkinghub.elsevier.com/retrieve/pii/S0951832011001335> [Accessed November 17, 2012].
- Clark, J. & Daigle, G., 1997. The Importance of Simulation Techniques in ITS Research and Analysis. In *1997 Winter simulation Conference*. pp. 1236–1243. Available at: <http://dl.acm.org/citation.cfm?id=268766> [Accessed August 29, 2013].
- Coppit, D., Sullivan, K.J. & Dugan, J.B., 2000. Formal Semantics of Models for Computational Engineering: A Case Study on Dynamic Fault Trees. In *Proceedings 11th International Symposium on Software Reliability Engineering. ISSRE 2000*. IEEE Comput. Soc, pp. 270–282. Available at: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=885878>.
- Dheedan, A., 2012. Distributed On-Line Safety Monitor Based on Safety Assessment Model and Multi-Agent System. University of Hull. Available at: <http://ethos.bl.uk/OrderDetails.do?uin=uk.bl.ethos.559107> [Accessed August 5, 2013].
- Dijkstra, E., 1961. Algol 60 Translation: An Algol 60 Translator For the X1 and Making a Translator for Algol 60, Report MR 35/61. Mathematisch Centrum, Amsterdam. Available at: <http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Algol+60+translation+An+algol+60+translator+for+the+x1+and+making+a+translator+for+algol+60#3> [Accessed July 30, 2013].
- Dinov, I., 2004. Continuous Random Variables and Probability Distributions. *Applied Probability and Statistics for Engineers*. Available at:

- http://www.stat.ucla.edu/~dinov/courses_students.dir/04/Spring/Stat110A.dir/STAT110A_notes.dir/PPCh04.pdf [Accessed October 28, 2013].
- DoD, 1998. Military Handbook: Electronic Reliability Design Handbook.
- Dugan, B.J., Bavuso, S.J. & Boyd, M.A., 1990. Fault Trees and Sequence Dependencies. In Reliability and Maintainability Symposium (RAMS). pp. 286–293.
- Dugan, J., 2001. Fault-Tree Analysis of Computer-Based Systems. Annual Reliability and Maintainability Symposium, (1), pp.1–83. Available at: <http://fault-tree.net/papers/dugan-comp-sys-fta-tutor.pdf> [Accessed July 29, 2013].
- Dugan, J.B., Bavuso, S.J. & Boyd, M. a., 1992. Dynamic Fault-Tree Models for Fault-Tolerant Computer Systems. IEEE Transactions on Reliability, 41(3), pp.363–377. Available at:
<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=159800>.
- Dugan, J.B. & Doyle, S.A., 1997. New Results in Fault-Tree Analysis.pdf. In Tutorial Notes of the Annual Reliability and Maintainability Symposium.
- Durga Rao, K., Gopika, V., Sanyasi Rao, V.V.S., Kushwaha, H.S., Verma, A.K. & Srividya, A., 2009. Dynamic Fault Tree Analysis Using Monte Carlo Simulation in Probabilistic Safety Assessment. Reliability Engineering & System Safety, 94(4), pp.872–883. Available at:
<http://linkinghub.elsevier.com/retrieve/pii/S0951832008002354> [Accessed November 17, 2012].
- Dutuit, Y. & Rauzy, A., 1996. A Linear-Time Algorithm to Find Modules of Fault Trees. IEEE Transactions on Reliability, 45(3), pp.422–425. Available at:
<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=537011>.
- Dutuit, Y. & Rauzy, A., 1997. Monte-Carlo Simulation to Propagate Uncertainties in Fault Trees Encoded by Means of Binary Decision Diagrams. In 1st International Conference on Mathematical Methods in Reliability, MMR'97. pp. 1–8. Available at: <http://www.lix.polytechnique.fr/~rauzy/publications/DutuitRauzySignoret1997-UncertaintyAnalysisInFaultTrees.pdf> [Accessed December 4, 2012].
- Edifor, E., Walker, M. & Gordon, N., 2012. Quantification of Priority-OR Gates in Temporal Fault Trees. Computer Safety, Reliability, and Security, (September). Available at: <http://www.springerlink.com/index/D2212861Q6068646.pdf> [Accessed November 12, 2012].
- Edifor, E., Walker, M. & Gordon, N., 2013. Quantification of Simultaneous-AND Gates in Temporal Fault Trees. Advances in Intelligent Systems and Computing, 224,

- pp.141–151. Available at: http://link.springer.com/chapter/10.1007/978-3-319-00945-2_13 [Accessed November 8, 2013].
- Edifor, E.E., Walker, M.D. & Gordon, N.A., 2013. Introducing Temporal Behaviour into Binary Decision Diagrams. In 4th IFAC Workshop on Dependable Control of Discrete Systems. pp. 7–12.
- Eid, M., 2011. A General Analytical Solution for the Occurrence Probability of a Sequence of Ordered Events Following Poisson Stochastic Processes. *THEORY & APPLICATIONS*, 2(21), pp.21–32. Available at: http://w.gnedenko-forum.org/Journal/2011/022011/RTA_2_2011-02.pdf [Accessed December 11, 2012].
- Ejlali, A. & Ghassem Miremadi, S., 2004. FPGA-Based Monte Carlo Simulation for Fault Tree Analysis. *Microelectronics Reliability*, 44(6), pp.1017–1028. Available at: <http://linkinghub.elsevier.com/retrieve/pii/S0026271404000769> [Accessed November 17, 2012].
- Esary, D. & Proschan, F., 1963. Coherent Structures with Non-Identical Components. *Technometrics*, 5(2), pp.191–209.
- Freund, J., 1962. *Mathematical Statistics*, New Jersey: Prince-Hall.
- Fussell, J.B., Aber, E.F. & Rahl, R.G., 1976. On the Quantitative Analysis of Priority-AND Failure Logic. *IEEE Transactions on Reliability*, R-25(5), pp.324–326. Available at: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5220025>.
- Gallier, J., 2010. *Discrete Mathematics*, New York: Springer Link.
- Garvey, P., 2008. *Analytical Methods for Risk Management: A Systems Engineering Perspective*, Boca Raton: CRC Press. Available at: [http://books.google.co.uk/books?hl=en&lr=&id=dCLXq78GDVgC&oi=fnd&pg=PR13&dq=analytical+methods+for+risk+management&ots=p0yM-oVKqo&sig=7qsq1IvbTEHMze2Reh2rNyDvP2w#v=onepage&q=analytical methods for risk management&f=false](http://books.google.co.uk/books?hl=en&lr=&id=dCLXq78GDVgC&oi=fnd&pg=PR13&dq=analytical+methods+for+risk+management&ots=p0yM-oVKqo&sig=7qsq1IvbTEHMze2Reh2rNyDvP2w#v=onepage&q=analytical+methods+for+risk+management&f=false).
- Geum, Y., Seol, H., Lee, S. & Park, Y., 2009. Application of Fault Tree Analysis to the Service Process: Service Tree Analysis Approach. *Journal of Service Management*, 20(4), pp.433–454. Available at: <http://www.emeraldinsight.com/10.1108/09564230910978520> [Accessed June 16, 2013].
- Gorski, J. & Wardzinski, A., 1996. Deriving Real-Time Requirements for Software from Safety Analysis. *Real-Time Systems*, 1996., ..., pp.9–14. Available at:

- http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=557782 [Accessed January 6, 2013].
- Gulati, R. & Dugan, J., 1997. A Modular Approach for Analyzing Static and Dynamic Fault Trees. In *Reliability and Maintainability Symposium*. pp. 57–63. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=571665 [Accessed December 10, 2012].
- Hamblin, C., 1962. Translation to and from Polish Notation. *The Computer Journal*, pp.210–213. Available at: <http://comjnl.oxfordjournals.org/content/5/3/210.short> [Accessed July 30, 2013].
- Hansen, K.M., Anders, P.R. & Stavridou, V., 1998. From Safety Analysis to Software Requirements. *IEEE Transactions on Software Engineering*, 24(7), pp.573–584. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=708570 [Accessed December 4, 2012].
- Henley, E.J. & Kumamoto, H., 1981. *Reliability Engineering and Risk Assessment*, Prince-Hall, Inc, New Jersey.
- Item, S., 2005. Fault Tree Mathematics Review. *Technical Support Notes*, p.10. Available at: [http://www.itemsoft.com/faq/Fault Tree Mathematics Review_R2.pdf](http://www.itemsoft.com/faq/Fault%20Tree%20Mathematics%20Review_R2.pdf) [Accessed September 16, 2013].
- Izvercian, M., Ivascu, L., Miclea, S. & Radu, A., 2012. Hazard Identification and Risk Assessment in Sustainable Enterprise. , pp.58–61.
- Kumamoto, H., Tanaka, K., Inoue, K. & Henley, E.J., 1980. State-Transition Monte Carlo for Evaluating Large, Repairable Systems. *IEEE Transactions on Reliability* *IEEE Transactions on*, R-29(5), pp.376–380. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5220888 [Accessed January 27, 2014].
- Lee, C., 1959. Representation of Switching Circuits by Binary-Decision Programs. *Bell System Technical Journal*, 38(4), pp.985–999. Available at: <http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Representation+of+Switching+Circuits+by+Binary-Decision+Programs#0> [Accessed August 19, 2013].
- Lee, W.S., Grosh, D.L., Tillman, F. a. & Lie, C.H., 1985. Fault Tree Analysis, Methods, and Applications - A Review. *IEEE Transactions on Reliability*, R-34(3), pp.194–203. Available at: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5222114>.

- Leveson, N., 2011. *Engineering a Safer World: Systems Thinking Applied to Safety* J. Moses, R. Neufville, M. Heitor, G. Morgan, E. Pate-Cornell, & W. Rouse, eds., Cambridge: The MIT Press. Available at:
http://books.google.com/books?hl=en&lr=&id=0gZ_7n5p8MQC&oi=fnd&pg=PR9&dq=Engineering+a+Safer+World+Systems+Thinking+Applied+to+Safety&ots=xNOAW3bFpl&sig=MIOCxvpTicc3EcYP20k7dW7HOec [Accessed October 4, 2013].
- Linial, N. & Nisan, N., 1990. Approximate Inclusion-Exclusion. *Combinatorica*, 10(4), pp.349–365. Available at:
<http://www.springerlink.com/index/F66M076T785578RX.pdf> [Accessed November 12, 2012].
- Liu, D., Xing, W., Zhang, C., Li, R. & Li, H., 2007. Cut Sequence Set Generation for Fault Tree Analysis. *Embedded Software and Systems LNCS*, 4523, pp.592–603. Available at: http://link.springer.com/chapter/10.1007/978-3-540-72685-2_55#page-2.
- Liu, D., Zhang, C., Xing, W., Li, R. & Li, H., 2007. Quantification of Cut Sequence Set for Fault Tree Analysis. *High Performance Computing and Communications LNCS*, 4782, pp.755–765. Available at:
http://link.springer.com/chapter/10.1007/978-3-540-75444-2_70#.
- Lozano, A. & Valiente, G., 2004. On the Maximum Common Embedded Subtree Problem for Ordered Trees. *String Algorithmics*. Available at:
<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.104.1438&rep=rep1&type=pdf> [Accessed June 20, 2013].
- Malhotra, M. & Trivedi, K.S., 1995. Dependability modeling using Petri-nets. *IEEE Transactions on Reliability*, 44(3), pp.428–440. Available at:
<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=406578>.
- Manno, G., Chiacchio, F., Compagno, L., D’Urso, D. & Trapani, N., 2012. MatCarloRe: An Integrated FT and Monte Carlo Simulink Tool for the Reliability Assessment of Dynamic Fault Tree. *Expert Systems with Applications*, 39(12), pp.10334–10342. Available at:
<http://linkinghub.elsevier.com/retrieve/pii/S0957417411016915> [Accessed November 17, 2012].
- Marquez, D., Neil, M. & Fenton, N., 2008. Solving Dynamic Fault Trees Using a New Hybrid Bayesian Network Inference Algorithm. In *16th Mediterranean Conference on Control and Automation*. pp. 609 – 614. Available at:

- http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4602222 [Accessed October 8, 2013].
- Marsan, M.A., 1989. Stochastic Petri Nets: An Elementary Introduction. Workshop on Applications and Theory in Petri Nets. Available at: <http://pdv.cs.tu-berlin.de/PMFE-SS2007/StochasticPetriNets.pdf> [Accessed April 23, 2013].
- Marseguerra, M., Zio, E., Devooght, J. & Labeau, P.E., 1998. A Concept Paper on Dynamic Reliability via Monte Carlo Simulation. *Mathematics and Computers in Simulation*, 47(2-5), pp.371–382. Available at: <http://linkinghub.elsevier.com/retrieve/pii/S0378475498001128>.
- Math.NET, 2013. Math.NET Numerics.
- Meedeniya, I., Moser, I., Aleti, A. & Grunske, L., 2011. Architecture-based reliability evaluation under uncertainty. Proceedings of the joint ACM SIGSOFT conference -- QoSA and ACM SIGSOFT symposium -- ISARCS on Quality of software architectures -- QoSA and architecting critical systems -- ISARCS - QoSA-ISARCS '11, p.85. Available at: <http://portal.acm.org/citation.cfm?doid=2000259.2000275>.
- Merle, G., 2010. Algebraic Modelling of Dynamic Fault Trees, Contribution to Qualitative and Quantitative Analysis. CACHAN, ÉNS DE. Available at: http://hal.inria.fr/docs/00/50/58/33/PDF/PhD_thesis_GM.pdf [Accessed July 29, 2013].
- Merle, G. & Roussel, J., 2007. Algebraic Modelling of Fault Trees with Priority AND Gates. In 1st IFAC Workshop on Dependable Control of Discrete Systems. pp. 175–180. Available at: <http://hal.archives-ouvertes.fr/hal-00350495/> [Accessed December 10, 2012].
- Merle, G., Roussel, J., Lesage, J. & Bobbio, A., 2009. Algebraic Expression of the Structure Function of a Subclass of Dynamic Fault Trees. Proceedings of the Available at: <http://hal.archives-ouvertes.fr/hal-00394459/> [Accessed July 29, 2013].
- Merle, G., Roussel, J., Lesage, J. & Bobbio, A., 2010. Probabilistic Algebraic Analysis of Fault Trees with Priority Dynamic Gates and Repeated Events. *IEEE Transactions on Reliability*, 59(1), pp.250–261. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5361394 [Accessed November 6, 2012].
- Merle, G., Roussel, J., Lesage, J. & Vayatis, N., 2010. Analytical Calculation of Failure Probabilities in Dynamic Fault Trees Including Spare Gates. *Reliability, Risk and*

- ..., (Esrel). Available at: <http://hal.archives-ouvertes.fr/hal-00516893/> [Accessed July 29, 2013].
- Merle, G., Roussel, J.-M. & Lesage, J.-J., 2013. Quantitative Analysis of Dynamic Fault Trees Based on the Structure Function. *Quality and Reliability Engineering International*. Available at: <http://doi.wiley.com/10.1002/qre.1487> [Accessed October 8, 2013].
- Montani, S. & Portinale, L., 2006. Automatically Translating Dynamic Fault Trees into Dynamic Bayesian Networks by Means of a Software Tool. In *The First International Conference on Availability, Reliability and Security, ARES*. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1625390 [Accessed October 8, 2013].
- Montani, S., Portinale, L. & Bobbio, A., 2005. Dynamic Bayesian Networks for Modeling Advanced Fault Tree Features in Dependability Analysis. In *Proceedings of the 16th European Conference on Safety and Reliability*. pp. 1415–1422. Available at: <http://people.unipmn.it/stefania/papers-pdf/C45.pdf> [Accessed October 9, 2013].
- Naikan, V.N., 2009. *Reliability Engineering and Life Testing*, New Delhi: PHI Learning Private Limited.
- O'Connor, A.N., 2011. *Probability Distributions Used in Reliability Engineering*, Reliability Information Analysis Center Reliability Information Analysis Center.
- Ong, M.-S. & Coiera, E., 2010. Safety Through Redundancy: A Case Study of in-Hospital Patient Transfers. *Quality & safety in health care*, 19(5), p.e32. Available at: <http://www.ncbi.nlm.nih.gov/pubmed/20671076> [Accessed June 15, 2014].
- OUP, 2013. *Reliable*. *Oxford English Dictionary*. Available at: <http://www.oed.com/view/Entry/161905?redirectedFrom=reliable#eid> [Accessed August 5, 2013].
- Palshikar, G.K., 2002. Temporal Fault Trees. *Information and Software Technology*, 44(3), pp.137–150. Available at: <http://linkinghub.elsevier.com/retrieve/pii/S0950584901002233>.
- Papadopoulos, Y., McDermid, J., Sasse, R. & Heiner, G., 2001. Analysis and Synthesis of the Behaviour of Complex Programmable Electronic Systems in Conditions of Failure. *Reliability Engineering & System Safety*, 71(3), pp.229–247. Available at: <http://linkinghub.elsevier.com/retrieve/pii/S0951832000000764>.
- Papoulis, A. & Pillai, U., 2002. *Probability, Random Variable and Stochastic Process* 4th ed., New York: McGraw-Hill.

- Pasquale, S.J., 2010. Teaching with Simulation. In W. B. Jeffries & N. Huggett, K, eds. *An Introduction to Medical Teaching*. New York: Springer, pp. 79–81. Available at:
<http://books.google.com/books?hl=en&lr=&id=U8AXD1QDO7AC&oi=fnd&pg=PR5&dq=An+Introduction+to+Medical+Teaching&ots=5GnnqUSr3z&sig=ZursHLbF0BnrDVzrJ2Rg0yLjVio> [Accessed August 29, 2013].
- Pearl, J., 1985. Bayesian Networks A Model of Self Activated Memory. In 7th Annual Conference of the Cognitive Science Society.
- Peterson, J.L., 1977. Petri Nets. *Computing Surveys*, 9(3), pp.223–252. Available at:
<http://www.rose-hulman.edu/Users/faculty/young/CS-Classes/csse373/Spring2009/Resources/peterson77.pdf>.
- Pukite, J. & Pukite, P., 1998. *Modelling for Reliability Analysis*, New York: Wiley-IEEE Press.
- Rauzy, A., 1993. *New Algorithms for Fault Trees Analysis*. *Reliability Engineering & System Safety*. Available at:
<http://www.sciencedirect.com/science/article/pii/095183209390060C> [Accessed July 31, 2013].
- Rocco, C. & Muselli, M., 2004. A Machine Learning Algorithm to Estimate Minimal Cut and Path Sets from a Monte Carlo Simulation. In *Probabilistic Safety Assessment and Management (PSAM7–ESREL '04): Proceedings of the 7th International Conference on Probabilistic Safety Assessment and Management*. pp. 3142–3147. Available at: <http://www.ice.ge.cnr.it/~muselli/papers/esrel04.pdf> [Accessed December 13, 2012].
- Rozenblit, J.W., 2001. System Design: A Simulation Modeling Framework. In H. S. Sarjoughian & F. Cellier, eds. *Discrete Event Modeling and Simulation Technologies: A Tapestry of Systems and AI-Based Theories and Methodologies*. Berlin: Springer-Verlag, pp. 107–110.
- Sadou, N. & Demmou, H., 2009. Reliability Analysis of Discrete Event Dynamic Systems with Petri Nets. *Reliability Engineering & System Safety*, 94(11), pp.1848–1861. Available at:
<http://linkinghub.elsevier.com/retrieve/pii/S0951832009001331> [Accessed October 9, 2013].
- Salem, A., Muller, A. & Weber, P., 2006. Dynamic Bayesian Networks in System Reliability Analysis. In 6th IFAC Symposium on Fault Detection, Supervision and

- Safety of Technical Processes. Available at: <http://hal.archives-ouvertes.fr/hal-00092032/> [Accessed October 8, 2013].
- Schneeweiss, W.G., 1989. *Boolean Functions with Engineering Applications and Computer Programs*, New York: Springer-Verlag.
- Sherstov, A. a., 2009. Approximate Inclusion-Exclusion for Arbitrary Symmetric Functions. *Computational Complexity*, 18(2), pp.219–247. Available at: <http://link.springer.com/10.1007/s00037-009-0274-4> [Accessed August 19, 2013].
- Shooman, M., 2002. *Reliability of Computer Systems and Networks*, New York: John Wiley and Sons.
- Sinnamon, R., 1996. Binary Decision Diagrams for Fault Tree Analysis. Available at: <https://dspace.lboro.ac.uk/dspace-jspui/handle/2134/7424> [Accessed December 17, 2012].
- Sinnamon, R. & Andrews, J., 1998. Improved Efficiency in Qualitative Fault Tree Analysis. *Quality and Reliability Engineering International*, 13, pp.293–298. Available at: [http://onlinelibrary.wiley.com/doi/10.1002/\(SICI\)1099-1638\(199709/10\)13:5<293::AID-QRE110>3.0.CO;2-Y/abstract](http://onlinelibrary.wiley.com/doi/10.1002/(SICI)1099-1638(199709/10)13:5<293::AID-QRE110>3.0.CO;2-Y/abstract) [Accessed January 18, 2013].
- Sinnamon, R.M. & Andrews, J.D., 1996. Fault Tree Analysis and Binary Decision Diagrams. *Proceedings of 1996 Annual Reliability and Maintainability Symposium*, pp.215–222. Available at: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=500665>.
- Sknourilova, P. & Bris, R., 2008. Coloured Petri nets and a dynamic reliability problem. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, 222(4), pp.635–642. Available at: <http://pio.sagepub.com/lookup/doi/10.1243/1748006XJRR155> [Accessed October 9, 2013].
- Stapelberg, F.R., 2009. *Handbook of Reliability, Availability, Maintainability and Safety in Engineering Design*, London: Springer.
- Sundararajan, C., 1991. *Guide To Reliability Engineering Data, Analysis, Applications, Implementation, and Management*, New York: Van Nostrand Reinhold.
- Tang, Z. & Dugan, J.B., 2004. Minimal Cut Set/Sequence Generation for Dynamic Fault Trees. *Annual Symposium Reliability and Maintainability, 2004 - RAMS*, pp.207–213. Available at: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1285449>.

- Terejanu, G.A., 2013. Tutorial on Monte Carlo Techniques. Tutorials, pp.1–15. Available at: <http://www.cse.sc.edu/~terejanu/files/tutorialMC.pdf> [Accessed October 2, 2013].
- Torres-Toledano, J. & Sucar, L., 1998. Bayesian Networks for Reliability Analysis of Complex Systems. Progress in Artificial Intelligence LNCS, 1484, pp.195–206. Available at: http://link.springer.com/chapter/10.1007/3-540-49795-1_17 [Accessed October 9, 2013].
- Trubiani, C., Meedeniya, I., Cortellessa, V., Aleti, A. & Grunske, L., 2013. Model-based performance analysis of software architectures under uncertainty. Proceedings of the 9th international ACM Sigsoft conference on Quality of software architectures - QoSA '13, p.69. Available at: <http://dl.acm.org/citation.cfm?doid=2465478.2465487>.
- USNRC, 2013. Backgrounder on Fire Protection for Nuclear Power Plants. Facts Sheets, p.1. Available at: <http://www.nrc.gov/reading-rm/doc-collections/factsheets/fire-protection-fs.html> [Accessed September 23, 2013].
- Vanany, I., Zailani, S. & Pujawan, N., 2009. Supply Chain Risk Management: Literature Review and Future Research. ... and Supply Chain Management (..., 2(March), pp.16–33. Available at: <http://www.igi-global.com/article/supply-chain-risk-management/2514> [Accessed July 17, 2013].
- Vesely, W.E., Davis, T.C., Denning, R.S. & Saltos, N., 1986. Measures of Risk Importance And Their Applications. NUREG/CR-3385, (May). Available at: <http://pbadupws.nrc.gov/docs/ML0716/ML071690031.pdf> [Accessed November 20, 2013].
- Vesely, W.E., Goldberg, F.F., Roberts, N.H. & Haasl, D.F., 1981. Fault Tree Handbook, Washington DC: US Nuclear Regulatory Commission. Available at: <http://books.google.com/books?hl=en&lr=&id=x9t9qjLFm9sC&oi=fnd&pg=PA1&dq=Fault+Tree+Handbook&ots=1-fqAnL4Hw&sig=dmtqp13m3mJFt598yllUQ6HqXCg> [Accessed February 27, 2013].
- Vesely, W.E., Stamatelatos, M., Dugan, J.B., Fragola, J., Minarick, J. & Railsback, J., 2002. Fault Tree Handbook with Aerospace Applications, Washington DC: NASA Office of Safety and Mission Assurance.
- Villen-altamirano, M. & Villen-altamirano, J., 1994. RESTART: A Straightforward Method for Fast Simulation of Rare Events. In Simulation Conference pp. 282–

289. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=717150 [Accessed October 21, 2013].
- Walker, M., 2009. Pandora: A Logic for the Qualitative Analysis of Temporal Fault Trees. University of Hull. Available at: <http://ethos.bl.uk/OrderDetails.do?uin=uk.bl.ethos.518653> [Accessed November 11, 2012].
- Walker, M., Bottaci, L. & Papadopoulos, Y., 2007. Compositional Temporal Fault Tree Analysis. *Computer Safety, Reliability, and Security LNCS*, 4680, pp.106–119. Available at: http://link.springer.com/chapter/10.1007/978-3-540-75101-4_12 [Accessed August 29, 2013].
- Walker, M. & Papadopoulos, Y., 2006. Pandora : The Time of Priority-AND Gates. In *12th IFAC International Symposium on Information Control Problems in Manufacturing*. Available at: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.108.5165> [Accessed September 27, 2013].
- Walker, M. & Papadopoulos, Y., 2007. Pandora 2: The Time of Priority-OR Gates. In *IFAC Workshop on Dependable Control of Discrete Event Systems*. Available at: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.108.5165> [Accessed November 11, 2012].
- Walker, M. & Papadopoulos, Y., 2008. Synthesis and Analysis of Temporal Fault Trees with PANDORA: The Time of Priority AND Gates. *Nonlinear Analysis: Hybrid Systems*, 2(2), pp.368–382. Available at: <http://linkinghub.elsevier.com/retrieve/pii/S1751570X06000574> [Accessed December 10, 2012].
- Wang, H. & Pham, H., 1997. Survey of Reliability and Availability Evaluation of Complex Networks Using Monte Carlo Techniques. *Microelectronics Reliability*, 31(2), pp.187–209. Available at: <http://www.sciencedirect.com/science/article/pii/S0026271496000583> [Accessed August 26, 2013].
- Weber, P. & Jouffe, L., 2003. Reliability Modelling with Dynamic Bayesian Networks. In *In 5th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes*. Available at: http://hal.archives-ouvertes.fr/docs/00/12/84/75/PDF/Weber_2003_Safeprocess_version_hal.pdf [Accessed October 9, 2013].

- Weinzierl, S., 2000. Introduction to Monte Carlo methods. , pp.1–47. Available at: <http://arxiv.org/pdf/hep-ph/0006269v1.pdf> [Accessed October 25, 2013].
- Weisstein, E.W., 2013. Monte Carlo Method. MathWorld - A Wolfram Web Resource. Available at: <http://mathworld.wolfram.com/MonteCarloMethod.html> [Accessed October 25, 2013].
- Wolfram, 2011. Mathematica 8 for Students.
- Yevkin, O., 2010. An Improved Monte Carlo Method in Fault Tree Analysis. In Reliability and Maintainability Symposium (RAMS). pp. 1–5. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5447989 [Accessed August 27, 2013].
- Yi, L., Bo, W., Dong, L., Haitao, Y. & Fande, Y., 2013. Complete Temporal Rules for Cut Sequence Generation in Dynamic Fault Tree Analysis. In Proceedings of the World Congress on Engineering. pp. 903–908. Available at: http://www.iaeng.org/publication/WCE2013/WCE2013_pp903-908.pdf [Accessed September 24, 2013].
- Yuge, T. & Yanagi, S., 2008. Quantitative Analysis of a Fault Tree with Priority AND Gates. Reliability Engineering & System Safety, 93(11), pp.1577–1583. Available at: <http://linkinghub.elsevier.com/retrieve/pii/S0951832008000409> [Accessed December 10, 2012].
- Ziegenbein, A. & Baumgart, J., 2006. Supply Chain Risk Assessment - A Quantitative Approach. In Proceedings of the International Conference on Information Systems Logistics and Supply Chain (ILS 2006), Lyon. Available at: www.fucam.ac.be/redirect.php3?id=46721.
- Zio, E., Marella, M. & Podofillini, L., 2007. A Monte Carlo Simulation Approach to the Availability Assessment of Multi-State Systems with Operational Dependencies. Reliability Engineering & System Safety, 92(7), pp.871–882. Available at: <http://linkinghub.elsevier.com/retrieve/pii/S0951832006001207> [Accessed December 5, 2012].

APPENDIXES

APPENDIX 1: Monte Carlo Condition for Various MCSQs

Where X , Y and Z , are events, r is a random number to simulate the probability of an event and t is the system lifetime. $\&\&$ and $\|$ are the logical AND and OR respectively. F is the CDF and TTF retains its previously described meaning. It must be noted that these conditions are for exponential distributions only.

$X < Y$: Referred to as SimPAND

$$r_x \leq F(x, t) \ \&\& \ r_y \leq F(y, t) \ \&\& \ TTF(x, r_x, t) < TTF(y, r_y, t);$$

$X | Y$: Referred to as SimPOR

$$r_x \leq F(x, t) \ \&\& \ r_y \leq F(y, t) \ \&\& \ TTF(x, r_x, t) < TTF(y, r_y, t) \ \| \\ (r_x \leq F(x, t) \ \&\& \ r_y > F(y, t))$$

$X \&_d Y$: Referred to as SimSAND

$$r_x \leq F(x, t) \ \&\& \ r_y > F(y, t) \ \&\& \ r_y \leq F(y, t + d) \ \| \ (r_y \leq F(y, t) \ \&\& \\ r_x > F(x, t) \ \&\& \ r_x \leq F(x, t + d))$$

$X < Y < Z$

$$r_x \leq F(x, t) \ \&\& \ r_y \leq F(y, t) \ \&\& \ r_z \leq F(z, t) \ \&\& \ TTF(x, r_x, t) < \\ TTF(y, r_y, t) \ \&\& \ TTF(x, r_x, t) < TTF(z, r_z, t) \ \&\& \ TTF(y, r_y, t) < TTF(z, r_z, t)$$

$X < Y | Z$

$$(SimPAND(r_x, x, r_y, y, t) \ \&\& \ r_z \leq F(z, t) \ \&\& \ TTF(x, r_x, t) < \\ TTF(z, r_z, t)) \ \| \ (SimPAND(r_x, x, r_y, y, t) \ \&\& \ r_z > F(z, t))$$

$X < Y \& Z$

$$r_x \leq F(x, t) \ \&\& \ SimSAND(r_y, y, r_z, z, t, d) \ \&\& \\ TTF(x, r_x, t) < TTF(y, r_y, t) \ \&\& \ TTF(x, r_x, t) < TTF(z, r_z, t)$$

$X \& Y . Z$

$$SimSAND(r_x, x, r_y, y, t, d) \ \&\& \ r_z \leq F(z, t)$$

$X < Y + Z$

SimPAND(r_x, x, r_y, y, t) // $r_z \leq F(z, t)$

$X | Y + Z$

SimPOR(r_x, x, r_y, y, t) // $r_z \leq F(z, t)$

APPENDIX 2: Algorithm to Evaluate Top-Event

```
// token = basic events or operators or parenthesis
//convert MCSQ to RPN
while (MCSQ contains tokens)
    tkn ← Read leftmost token
    if (tkn is basic event) then
        Add failure probability of tkn to output queue
    end if
    if (tkn is operator) then
        op1 ← tkn
        while (top of stack is operator token, op2, AND op1 has less precedence than
op2)
            Pop op2 off stack unto output queue
        end while
        Push op1 unto the stack
    end if
    if (tkn is left parenthesis) then
        Push tkn unto the stack
    end if
    if (tkn is right parenthesis) then
        while (token at top of stack is not left parenthesis)
            Pop operators off stack unto output queue
        end while
        Pop left parenthesis from stack; not unto output queue
        if (stack is empty without finding left parenthesis) then
            Error: mismatched parenthesis
        end if
while (stack contains operators)
    if (operator on top of stack is parenthesis) then
        Error: mismatched parenthesis
    end if
end while
Pop operator unto output queue
rpn ← output queue

//Evaluate MCSQ RPN
while (rpn contains tokens)
    tkn ← read leftmost token
    if (tkn is an event) then
        push tkn unto stack
    else if (tkn is an operator) then
        op1 ← tkn
        if (stack has less than 2 values) then
```

```

error: insufficient input values in MCSQ
else
  pop top two values on stack
  switch (op1)
    case '&': temp = _SAND (num2, num1, d)
    case '<': temp = _PAND (num2, num1)
    case '|': temp = _POR (num2, num1)
    case '.': temp = _AND (num2, num1)
    case '+': temp = _OR (num2, num1)
    case else: Error: Unknown operator
  end switch
  push temp unto stack
end if
end if
end while
if (stack contains 1 value) then
  MCSQprobability ← value on top of the stack
else
  error: invalid input. Input has too many values

```

As previously discussed, the OR gate is the least in the order of precedence. This makes it easy to represent the top event in the form of sum of other gates at the top level in the resulting temporal fault tree. Using the Eqn (2) we present an algorithm for the evaluation of the top event as:

```

reliability ← 1
top-event ← 0
while (MCSQList contains MCSQ)
  mcsqProbability ← Next MCSQ probability //using algorithms above
  reliability ← reliability * (1 - mcsqProbability)
end while
top-event = 1 - reliability

```

APPENDIX 3: Preliminary List of CSQs

[I-SOL<Hi-SOF].Hi-SIF.Hi-SIF.[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].Hi-SEF.[Hi-SEF|Hi-SIF].[Hi-SEF|I-SCV].I-SOV.I-SOV

[I-SOL<Hi-SOF].Hi-SIF.Hi-SIF.[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].Hi-SEF.[Hi-SEF|Hi-SIF].[Hi-SEF|I-SCV].I-SOL.I-SOL

[I-SOL&Hi-SOF].Hi-SIF.Hi-SIF.[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].Hi-SEF.[Hi-SEF|Hi-SIF].[Hi-SEF|I-SCV].I-SOV.I-SOV

[I-SOL&Hi-SOF].[Hi-SEF|I-SCP].[Hi-SEF|Hi-SIF].[Hi-SEF|I-SCV].Hi-SOF.Hi-SOF.Hi-SIF.Hi-SIF.Hi-SEF.Hi-SEF.I-SOL.I-SOL

[Hi-SEF|I-SCP].[Hi-SEF|Hi-SOF].[Hi-SEF|Hi-SIF].Hi-SIF.Hi-SIF.[Hi-SEF|I-SCV].Hi-SEF.Hi-SEF.I-SOL.I-SOL

[Hi-SEF|I-SOV].[Hi-SEF|I-SOL].[Hi-SEF|I-SCP].Hi-SIF.Hi-SIF.[Hi-SEF|I-SCV].Hi-SEF.Hi-SEF.I-SOL.I-SOL

[Hi-SOF&I-SOL].Hi-SOF.Hi-SOF.[Hi-SEF|I-SOV].[Hi-SEF|I-SCP].[Hi-SEF|Hi-SIF].Hi-SIF.Hi-SIF.[Hi-SEF|I-SCV].Hi-SEF.Hi-SEF.I-SOL.I-SOL

[I-SOV<Hi-SOF].Hi-SIF.Hi-SIF.[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].Hi-SEF.[Hi-SEF|Hi-SIF].[Hi-SEF|I-SCV].I-SOV.I-SOV

Hi-SIF.Hi-SIF.Hi-SEF.Hi-SEF.I-SOL.I-SOL.[Hi-SOF&I-SOV].[Hi-SEF|I-SOL].[Hi-SEF|I-SCP].[Hi-SEF|Hi-SIF].[Hi-SEF|I-SCV]

[Hi-SEF|I-SOV].Hi-SEF.[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|Hi-SOF].[Hi-SEF|I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[I-SOL<Hi-SOF].[Hi-SOF<Hi-SEF].[Hi-SEF|I-SOV].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[I-SOL<Hi-SOF].[Hi-SOF<Hi-SEF].[Hi-SEF|I-SOV].[Hi-SEF<I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[Hi-SOF<Hi-SEF].[Hi-SEF|I-SOV].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[Hi-SOF<Hi-SEF].[Hi-SEF|I-SOV].[Hi-SEF<I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|I-SCV]

[I-SOV&Hi-SOF].Hi-SEF.[Hi-SEF|I-SOV].[Hi-SEF<I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF<Hi-SOF].[Hi-SEF|I-SCV]

[Hi-SOF&I-SOL].Hi-SEF.[Hi-SEF|I-SOV].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|I-SCV]

[Hi-SOF&I-SOL].Hi-SEF.[Hi-SEF|I-SOV].[Hi-SEF<I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|I-SCV]
Hi-SOF.Hi-SEF.Hi-SEF.Hi-SEF.[I-SOL<Hi-SEF].[Hi-SOF<I-SOL].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|Hi-SOF].[Hi-SEF|I-SCV]

[Hi-SEF|I-SOL].Hi-SEF.[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|Hi-SOF].[Hi-SEF|I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[Hi-SEF|I-SOL].[Hi-SEF<I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|Hi-SOF].[Hi-SEF|I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[I-SOL<Hi-SOF].[Hi-SOF<Hi-SEF].[Hi-SEF|I-SOL].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[I-SOL<Hi-SOF].[Hi-SOF<Hi-SEF].[Hi-SEF|I-SOL].[Hi-SEF<I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|I-SCV]

[I-SOL<Hi-SOF].Hi-SEF.[Hi-SOF<Hi-SEF].[Hi-SEF|I-SOL].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|I-SCV]

[I-SOL<Hi-SOF].Hi-SEF.[Hi-SOF<Hi-SEF].[Hi-SEF|I-SOL].[Hi-SEF<I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|I-SCV]

[I-SOL&Hi-SOF].Hi-SEF.[Hi-SEF|I-SOL].[Hi-SEF<I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF<Hi-SOF].[Hi-SEF|I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[Hi-SEF|I-SOL].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[Hi-SEF|I-SOL].[Hi-SEF<I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|I-SCV]

[I-SOV&Hi-SOF].Hi-SEF.[Hi-SEF|I-SOL].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|I-SCV]

[Hi-SOF&I-SOV].Hi-SEF.[Hi-SEF|I-SOL].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|I-SCV]

[Hi-SOF&I-SOV].Hi-SEF.[Hi-SEF|I-SOL].[Hi-SEF<I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|I-SCV]

[I-SOL<Hi-SEF].Hi-SEF.[I-SIV<I-SOL].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|Hi-SOF].[Hi-SEF|I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[I-SOL<Hi-SOF].[I-SOL<Hi-SEF].[I-SIV<I-SOL].[I-SIV<Hi-SOF].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|I-SCV]

[I-SOL<Hi-SOF].Hi-SEF.[I-SOL<Hi-SEF].[I-SIV<I-SOL].[I-SIV<Hi-SOF].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|I-SCV]

[I-SOL&Hi-SOF].Hi-SEF.[I-SOL<Hi-SEF].[I-SIV<I-SOL].[I-SIV<Hi-SOF].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[I-SOL<Hi-SEF].[I-SIV<I-SOL].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|I-SCV]

[I-SOL<Hi-SOF].Hi-SEF.[I-SIV&I-SOL].[I-SIV<Hi-SOF].[Hi-SEF<I-SOL].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|I-SCV]

[I-SOL&Hi-SOF].Hi-SEF.[I-SIV&I-SOL].[I-SIV&Hi-SOF].[Hi-SEF<I-SOL].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|I-SCV]

[Hi-SEF|I-SIV].Hi-SEF.[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|Hi-SOF].[Hi-SEF|I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[Hi-SEF<I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|Hi-SOF].[Hi-SEF|I-SCV]

[I-SOL<Hi-SOF].Hi-SEF.[Hi-SEF<I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|Hi-SOF].[Hi-SEF|I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[I-SOL<Hi-SOF].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[I-SOL<Hi-SOF].[Hi-SEF<I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|I-SCV]

[I-SOL<Hi-SOF].Hi-SEF.[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|I-SCV]

[I-SOL<Hi-SOF].Hi-SEF.[Hi-SEF<I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|I-SCV]

[I-SOL&Hi-SOF].Hi-SEF.[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|I-SCV]

[I-SOL&Hi-SOF].Hi-SEF.[Hi-SEF<I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[Hi-SEF<I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|I-SCV]

[I-SOL&Hi-SOF].[Hi-SEF|I-SOV].[Hi-SEF|I-SOL].[Hi-SEF<I-SIV].[Hi-SEF|I-SCP].[Hi-SEF<Hi-SOF].[Hi-SEF|I-SCV]

[Hi-SEF<I-SOL].[Hi-SEF<I-SIV].[Hi-SEF|I-SCP].[Hi-SEF|Hi-SOF].[Hi-SEF|I-SCV]

[Hi-SEF|I-SOV].[Hi-SEF|I-SOL].[Hi-SEF<I-SCP].[Hi-SEF|I-SCV]

I-SIV.[Hi-SEF|I-SOV].[Hi-SEF<I-SOL].[Hi-SEF|I-SCP].[Hi-SEF|I-SCV]

[I-SIL].[Hi-SEF|I-SOV].[Hi-SEF<I-SOL].[Hi-SEF|I-SCP].[Hi-SEF|I-SCV]
[Hi-SEF<I-SOL].[Hi-SEF<I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|I-SCV]

[I-SOL<Hi-SOF].[Hi-SOF<Hi-SEF].[Hi-SEF|I-SOV].[Hi-SEF|I-SOL].[Hi-SEF<I-SIV].
[Hi-SEF|I-SCP].[Hi-SEF|I-SCV]

[Hi-SOF&I-SOL].[Hi-SEF|I-SOV].[Hi-SEF|I-SOL].[Hi-SEF<I-SIV].[Hi-SEF|I-SCP].
[Hi-SEF<Hi-SOF].[Hi-SEF|I-SCV]

[I-SOV<Hi-SOF].[Hi-SOF<Hi-SEF].[Hi-SEF|I-SOV].[Hi-SEF|I-SOL].[Hi-SEF<I-SIV].
[Hi-SEF|I-SCP].[Hi-SEF|I-SCV]

[I-SOV&Hi-SOF].[Hi-SEF|I-SOV].[Hi-SEF|I-SOL].[Hi-SEF<I-SIV].[Hi-SEF|I-SCP].
[Hi-SEF<Hi-SOF].[Hi-SEF|I-SCV]

[Hi-SOF&I-SOV].[Hi-SEF|I-SOV].[Hi-SEF|I-SOL].[Hi-SEF<I-SIV].[Hi-SEF|I-SCP].
[Hi-SEF<Hi-SOF].[Hi-SEF|I-SCV]

[Hi-SOF|I-SOV].[Hi-SOF<I-SOL].[Hi-SEF|I-SOV].[Hi-SEF<I-SOL].[Hi-SEF|I-SCP].
[Hi-SEF|I-SCV]

[I-SIL<I-SOV].[Hi-SEF|I-SOV].[Hi-SEF|I-SOL].[Hi-SEF<I-SIV].[Hi-SEF|I-SCP].[Hi-SEF|I-SCV]

[I-SIL<I-SOV].[Hi-SEF<I-SOL].[Hi-SEF<I-SIV].[Hi-SEF|I-SCP].[Hi-SEF|I-SCV]

[Hi-SEF<I-SOL].[Hi-SEF<I-SIV].[Hi-SEF|I-SCP].[Hi-SEF|Hi-SIF].[Hi-SEF|I-SCV]

[I-SOV&Hi-SOF].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF<I-SCP].[Hi-SEF|Hi-SIF].
[Hi-SEF|I-SCV]

[I-SOV&Hi-SOF].[Hi-SEF<I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|Hi-SIF].
[Hi-SEF|I-SCV]

[I-SIL<Hi-SIF].[Hi-SEF<I-SOL].[Hi-SEF<I-SIV].[Hi-SEF|I-SCP].[Hi-SEF|I-SCV]

[I-SIL&Hi-SIF].[Hi-SEF<I-SOL].[Hi-SEF<I-SIV].[Hi-SEF|I-SCP].[Hi-SEF|I-SCV]

[Hi-SEF|I-SOV].Hi-SEF.[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|Hi-SOF].
[Hi-SEF<I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[I-SOL<Hi-SOF].[Hi-SOF<Hi-SEF].[Hi-SEF|I-SOV].[Hi-SEF|I-SIV].
[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF<I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[Hi-SOF<Hi-SEF].[Hi-SEF|I-SOV].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].
[Hi-SEF|I-SCP].[Hi-SEF<I-SCV]

[Hi-SOF&I-SOL].Hi-SEF.[Hi-SEF|I-SOV].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].
[Hi-SEF<I-SCV]

[Hi-SEF|I-SOL].Hi-SEF.[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|Hi-SOF].
[Hi-SEF<I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[I-SOL<Hi-SOF].[Hi-SOF<Hi-SEF].[Hi-SEF|I-SOL].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF<I-SCV]

[I-SOL<Hi-SOF].Hi-SEF.[Hi-SOF<Hi-SEF].[Hi-SEF|I-SOL].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF<I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[Hi-SEF|I-SOL].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF<I-SCV]

[Hi-SOF&I-SOV].Hi-SEF.[Hi-SEF|I-SOL].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF<I-SCV]

[I-SOL<Hi-SEF].Hi-SEF.[I-SIV<I-SOL].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|Hi-SOF].[Hi-SEF<I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[I-SOL<Hi-SOF].[I-SOL<Hi-SEF].[I-SIV<I-SOL].[I-SIV<Hi-SOF].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF<I-SCV]

[I-SOL<Hi-SOF].Hi-SEF.[I-SOL<Hi-SEF].[I-SIV<I-SOL].[I-SIV<Hi-SOF].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF<I-SCV]

[I-SOL&Hi-SOF].Hi-SEF.[I-SOL<Hi-SEF].[I-SIV<I-SOL].[I-SIV<Hi-SOF].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF<I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[I-SOL<Hi-SEF].[I-SIV<I-SOL].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF<I-SCV]

[I-SOL<Hi-SOF].Hi-SEF.[I-SIV&I-SOL].[I-SIV<Hi-SOF].[Hi-SEF<I-SOL].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF<I-SCV]

[I-SOL&Hi-SOF].Hi-SEF.[I-SIV&I-SOL].[I-SIV&Hi-SOF].[Hi-SEF<I-SOL].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF<I-SCV]

[Hi-SEF|I-SIV].Hi-SEF.[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|Hi-SOF].[Hi-SEF<I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[I-SOL<Hi-SOF].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF<I-SCV]

[I-SOL<Hi-SOF].Hi-SEF.[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF<I-SCV]

[I-SOL&Hi-SOF].Hi-SEF.[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF<I-SCV]

[I-SOV<Hi-SOF].Hi-SEF.[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF<I-SCV]

[Hi-SEF|I-SOV].[Hi-SEF|I-SOL].[Hi-SEF|I-SCP].[Hi-SEF<I-SCV]

[I-SOV&Hi-SOF].[Hi-SEF|I-SIV].[Hi-SEF|I-SIL].[Hi-SEF|I-SCP].[Hi-SEF|Hi-SIF].[Hi-SEF<I-SCV]

I-SIL.[I-SOV&I-SIL].I-SIL.I-SOV.I-SOV

I-SOV.[I-SOV&I-SIV].I-SOV.I-SIV.I-SIV

I-SOL.[I-SOL&I-SIV].I-SOL.I-SIV.I-SIV

[I-SIV<I-SOV].[I-SIV|I-SOL]

[I-SIV|I-SOV].[I-SIV<I-SOL]

[I-SIL<I-SOV].[I-SIL|I-SOL]

[Hi-SIF|I-SIL].Hi-SIF.I-CSP.I-SOV.I-SOV

I-CRL.Hi-SIF.Hi-SIF.I-SOV.I-SOV

Hi-SIF.Hi-SIF.I-SOV.I-SOV.I-CSV

Hi-SIF.Hi-SIF.I-CSP.I-SOL.I-SOL

Hi-SIF.Hi-SIF.I-SOL.I-SOL.I-CSV

I-SCP.I-CSP

I-CRL.I-SCP

I-SCP.I-CSV

I-CSP.I-SOL.I-SIL

I-CRL.I-SOL.I-SIL

I-SOL.I-SIL.I-CSV

I-CSP.[Hi-SOF<I-SOV].[Hi-SOF|I-SOL]

I-CRL.[Hi-SOF<I-SOV].[Hi-SOF|I-SOL]

I-CSV.[Hi-SOF<I-SOV].[Hi-SOF|I-SOL]

I-CSP.[Hi-SOF|I-SOV].[Hi-SOF<I-SOL]

I-CRL.[Hi-SOF|I-SOV].[Hi-SOF<I-SOL]

I-CSV.[Hi-SOF|I-SOV].[Hi-SOF<I-SOL]

I-CSP.I-SCV

I-CRL.I-SCV

I-CSV.I-SCV

APPENDIX 4: Supply Chain Risk Management Using Pandora

In this appendix, Pandora's logic gates are used to model some supply chain scenarios. These are then used to evaluate the reliability of a hypothetical supply chain. It must be noted that this appendix is the author's concept of how Pandora can be used in analysing the reliability of supply chain. Although there may be existing literature covering some parts of this contribution (especially, with classical Boolean Gates), no literature review is made – the entire content, except for the case study, is solely the author's work.

Modelling of Scenarios

First of all, some key terminologies to be used in this appendix are redefined to suit the context in which they are used.

Entity: A person, process, department, organisation or equipment forming part of a supply chain; these entities contribute to the success of the supply chain.

System: A group of interconnected entities operating to achieve a common goal.

Event: A binary outcome of the operation of an entity. This could be either a success or a failure.

Failure: A system or entity's inability to deliver its intended function.

Reliability: A system or entity's ability to deliver its intended function withig a specified time in the conditions it was designed to function.

Before Pandora can be used in analysing fault trees of supply chains, each gate (both temporal and static) is described in the "supply chain" context to model various scenarios. All gates maintain their symbols and representations described in earlier chapters of this thesis.

AND Gate

The AND gate logic describes the scenario where one or more events in a supply chain occur. Meaning, the output event of an AND gate is triggered if all its input events occur. For example, a manufacturing firm, MF , which has more than one channel, $C1$, $C2$... CN , of getting raw materials will fail to function if all its channels fail. That is:

$$MF_{fails} = C1_{fails} \text{ AND } C2_{fails} \text{ AND } \dots \text{ AND } CN_{fails}$$

OR Gate

The OR gate represents the scenario where the failure of at least one event in a supply chain can lead to the failure. Meaning, the output event of an OR gate is triggered if at least one of its input events occur. For example, a manufacturing firm, MF , will fail if all its power plants, $F1$, fail or if all its workers, $F2$, fail to turn up for work or if all its channels of raw materials, $F3$, fail. Assuming all event failures leading the failure of MF are $F1, F2 \dots FN$ then,

$$MF_{fails} = F1_{fails} \text{ AND } F2_{fails} \text{ AND } \dots \text{ AND } FN_{fails}$$

Priority-AND (PAND) Gate

The PAND gate is used to model situations where the occurrence of an event strictly before the occurrence of another event will trigger the occurrence of another event. Meaning, an output event is fired when its input events occur in a sequence – strictly one after another. For example, consider a manufacturing firm, MF , with two production lines, $P1$ and $P2$. $P1$ is a primary production line and $P2$ is a standby production line. If $P1$ fails before $P2$, there is a chance that immediately $P1$ fails, $P2$ will quickly get activated into production so the entire production system does not necessarily fail. However, if $P2$ has already failed before $P1$, then when $P1$ fails, the entire production system fails. The PAND gate is used to represent the latter – $P2$ PAND $P1$.

Parameterised Simultaneous-AND (pSAND) Gate

The pSAND gate is used to represent the scenario where the occurrence of two or more events within a relatively short duration of time will trigger the occurrence of another event. Therefore, with the pSAND gate, an output event is triggered if its input events occur within a small interval of duration. For example, an automotive company, A , has only one source of steel - from a firm C . A has an outsourcing department, D , that is responsible for ordering steel from alternative sources should C fail. If D fails to source for steel within seven working days (due to employee strike etc.), another department X in A takes on the duty of ordering for steel; during this time, the production (of vehicle bodies), P , comes to a halt. The pSAND gate is the logic for representing this scenario where P fails when C and D fail within an interval of duration. Therefore, using the notations described earlier,

$$P_{fails} = C_{fails} \text{ pSAND}_{7\text{days}} D_{fails}$$

Priority-OR (POR) Gate

The Priority-OR gate is used to represent the scenario where an event occurs strictly before its subsequent event or the earlier event occurs and the later does not. Meaning, for two events, A and B , the output event of a POR gate, C , is triggered if its first input event A occurs strictly before its second input event B or A occurs but B does not. For example, a medium size shoe making company, C , gets its leather from a source S . Assuming one of C 's employees E , has two primary responsibilities: ordering leather from alternative sources if S fails to deliver leather on time and cutting the leather into shapes for making preordered custom shoes for customers. In this case, the production of custom made shoes, P , will come to a halt if E fails before S or E fails and S does not. Therefore,

$$P_{fails} = E_{fails} \text{ POR } S_{fails}$$

Case Study: Suces Fruit Company (SFC)

This case study demonstrates how Pandora, a temporal fault tree analysis tool, can be used to evaluate the reliability of a hypothetical supply chain model and to determine the critical aspects of the model. The diagram in Figure APX4-1 is a model of a fruit juice manufacturing company called Suces Fruit Company (SFC). SFC is a large United Kingdom (UK)-based fruit juice manufacturing company. Its primary function is to import raw fruits from suppliers, extract juices from these raw fruits and package them for retailers to sell to consumers. Suces has international suppliers in addition to what Suces gets from its local suppliers. It imports the two most patronised fruits in the UK fruit juice market – oranges and apples – from international sources. This is so because the UK local manufacturers do not produce enough oranges and apples to satisfy the demands of UK's thirsty market.

Suces has three big plants across the UK: one in the southeast (SE), another in the northeast (NE) and the third in the mid-west (MW) regions. Suces has three main processes – pre-extraction (PE), extraction and packaging (EP) and distribution (DC). During the pre-extraction process, the imported raw fruits undergo separation and washing; where good fruits are separated from bad ones and are cleaned with water and other chemicals to remove all forms of germs and bacteria. The extraction process, involves the actual extraction of juices, straining, filtration, blending and pasteurisation. During

the packaging process, the fruit juice is put into appropriate packages, sealed, sterilised, labelled and packaged into boxes. Suces has a three extraction and three packaging plants: one of each at a production site. However, to save cost, it has only one pre-extraction plant located in the mid-west, which supplies prepared raw fruits to all the extraction and packaging plants. Figure APX4-1 is an abstract supply chain structure of the plant distribution over the UK.

Suces produces only orange and apple juices so it imports only raw oranges and apples. It imports huge quantities of its oranges and apples from Agro Exporters (AE) in Spain. and Yaba Exporters (YE) based in Italy. Argo and Yaba are Suces' primary sources of raw materials. Smot Exporters (SE) in Mexico is Suces' secondary source of raw oranges and apples. Because of the cost of shipping, Suces rarely imports from Smot. It only does so when there is an unusual increase in demand for both orange and apple juice – especially during festive seasons. All imported raw fruits are directed to the PE for cleaning and screening. Once the pre-extraction processes are completed, the prepared raw fruits are sent to all three extraction and packaging plants; of which one is onsite, close to the pre-extraction plant. Each EP plant has a DC, which serves as a warehouse and a distribution point of finished products to retailers (RT). These retailers are supermarkets who make Suces' finished products available to consumers. Figure APX4-2 is an abstract graphical model of Suces' supply and distribution network.

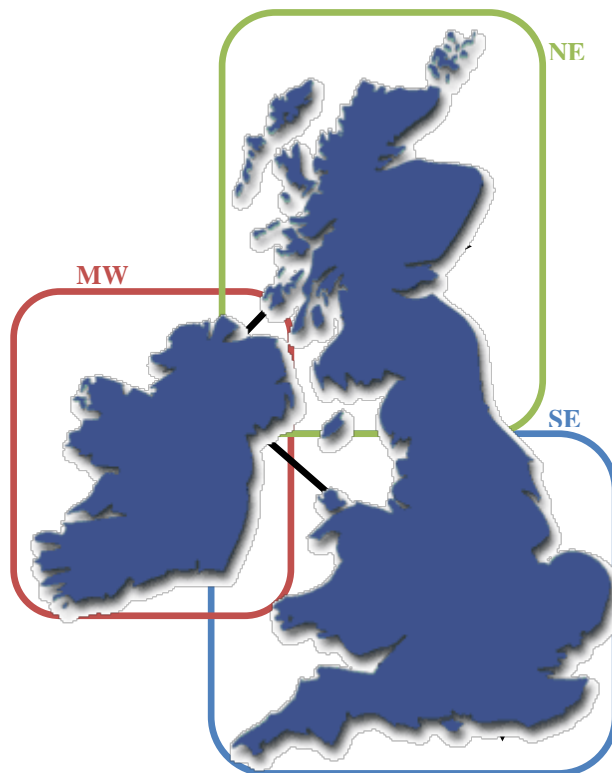


Figure APX4- 1: Suces regional structure

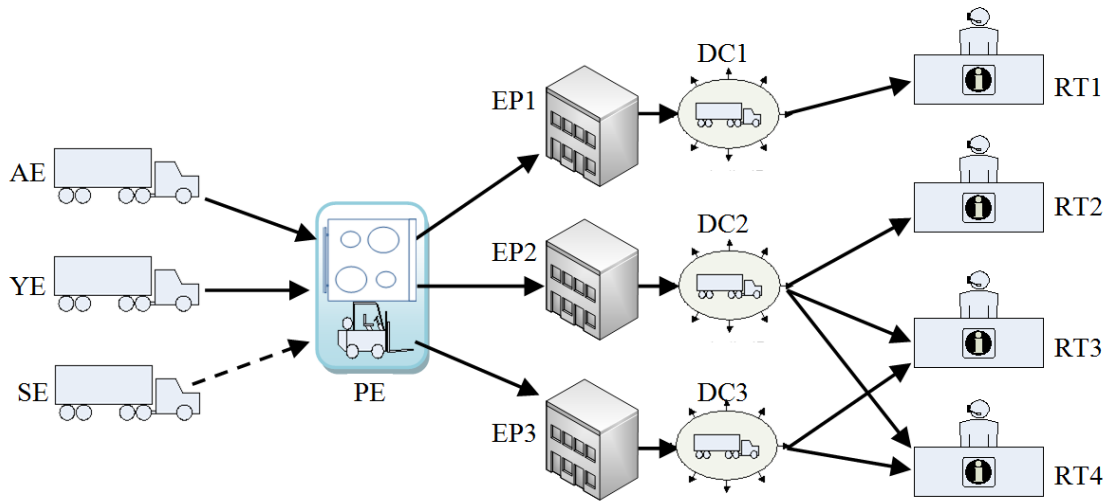


Figure APX4- 2: Suces fruit company supply chain model

Suces has an Inventory Department, ID, responsible for monitoring the stock of imported raw fruits. ID ensures that:

1. there is a constant supply of raw fruits to PE. If the raw fruits are 20% less than PE's storage capacity, a request is placed for AE and YE to supply the 80% shortage. Originally, each is required to supply 50% of the shortage. However, if one cannot meet up with the demand, the other is asked to supply what was lacking.
2. if AE and YE supply over 60% of the required 80% shortage, local suppliers will be contacted to supply the remaining 20% else Smot will be contacted.
3. if both of them fail to supply the needed 80% of shortage, Smot, the secondary supplier, is asked to satisfy shortage. It is very unlikely that Smot will be unable to satisfy a request. However, if Smot is not able to satisfy the 80% shortage, local (UK-based) fruit farmers, who produce very small quantities of oranges or apples will be contacted.

Suces also has a Distribution Department (DD) which is responsible for the even distribution of finished products to the retailers who sell the products to consumers across the three regions.

The entire logistic network model of Suces is considered coherent: no part of the system can be improving while others are failing. For example, It is assumed that if the total output production of Suces is decreasing due to the low supply of orange, there will be no increase in apple juice production to make up for the loss.

Qualitative Analysis

What can cause a shortage in finished products: O-juice

$$O-PE = PE + AE \& YE \& SE + SE < (AE \cdot YE)$$

$$O-EP1 = EP1 + O-PE$$

$$O-EP2 = EP2 + O-PE$$

$$O-EP3 = EP3 + O-PE$$

$$O-DC1 = DC1 + O-EP1$$

$$O-DC2 = DC2 + O-EP2$$

$$O-DC3 = DC3 + O-EP2 \cdot O-EP3$$

$$O-RT1 = O-DC1$$

$$O-RT2 = O-DC2$$

$$O-RT3 = O-DC2 \cdot O-DC3$$

$$O-RT4 = O-DC3 \cdot O-DC4$$

$$O\text{-juice} = O-RT1 \cdot O-RT2 \cdot O-RT3 \cdot O-RT4$$

Finally,

$$\begin{aligned} O\text{-juice} = & DC1 \cdot DC2 \cdot DC3 + DC1 \cdot DC3 \cdot EP2 + DC1 \cdot EP2 \cdot EP3 + \\ & DC2 \cdot DC3 \cdot EP1 + DC3 \cdot EP1 \cdot EP2 + EP1 \cdot EP2 \cdot EP3 + PE \\ & + [AE\&YE] \cdot [AE\&SE] \cdot [SE\&YE] + AE \cdot [SE < YE] + [SE < AE] \cdot YE \end{aligned}$$

The above expression for o-juice are the MCSQs. From the MCSQs, it is obvious that PE (pre-extraction) is the only critical part of the system. It is critical because when it fails, the entire system fails. Hence Suces' management must spend/allocate resources in keeping this section running at all times.

Using the techniques proposed in this thesis, the quantitative analysis (top-event and importance measures) can be determined if some failure data are provided for the events. From the quantitative measure, importances can be attached to each individual MCSQ. These will further inform Suces' management contributions of MCSQ leading to the top-event.

GLOSSARY

Archimedes

An inductive methodical technique for analysing temporal fault trees. It does so by converting fault trees into an alternative structure by enumerating all possibilities of cut sequences occurring and then generating possible sequences for a set of events.

Balanced Sequence

A string of zeroes and ones that represent a fault tree. The fault tree is traversed depth-first from left to right, each descent is a zero and each ascent is a one. Balanced sequences usually contains other balanced sequences: these can be considered as sub-trees.

Basic Event

A single component failure that contributes to the failure of another component. It can have not causative component failures.

Binary Decision Diagram (BDD)

A directed acyclic graph. Static fault trees can be translated into BDDs and analysed quantitatively without qualitative analysis. The quantitative analysis can yield both top-event probabilities and importance measures.

Birnbaum Measure (BM) of Importance

It determines the sensitivity of the top event probability with respect to some given events. In other words, measures the rate of change in the top-event probability with respect to the changes of the probability of a specific event.

Component

A subsystem or integral part of a system that contributes, usually in addition and relation to other components, to the overall functionality of the system.

Cold Spare (CSP)

A redundant or standby component that replaces a primary components when it (primary component) fails. It is used in DFT analysis.

Continuous Random Variable

Random variable defined over a continuum which is supposedly uncountable/non-discrete.

Cut Set

A combination of logically related basic events that propagate to cause a total system failure or top event.

Dynamic Fault Tree (DFT)

A fault tree that considers the temporal or dynamic behaviours present in systems by the use of dynamic gates – FDEP, CSP, PAND, SEQ. The temporal/dynamic behaviours include the sequential or order in which events occur. DFTs are usually analysed quantitatively.

Error

A mistake caused by the manifestation of a fault.

Euripides

A law-based deductive technique for logically analysing temporal fault trees. It does so by the use of four processes: binarboreal, flattening, encapsulation and minimization.

Event

A binary outcome of a component's functionality: whether it has failed or it is operational.

Exclusive Priority-AND (ePAND)

A logical gate that is triggered when an event occurs strictly before another event; its input events cannot occur at the same time.

Failure

A system's deviation from its specified performance; meaning, the system is unable to deliver the function(s) it was designed to deliver.

Failure Rate

The rate at which the failure of a component/system fails per unit of time. Meaning, the number of failures of a system per unit of time.

Fault

An abnormal or erroneous state of a system.

Fault Tree Analysis (FTA)

A deductive technique of analysing a system by considering how the basic combinations of components of the system propagate to cause a system failure.

Functional Dependency (FDEP)

A logical gate used to model Common Cause Failures (CCF) of events. CCF occurs when the occurrence of a single event failure triggers the failure occurrence of other dependent events almost at the same time.

Fussell-Vesely (FV) Importance

A measure of the contribution of a particular basic event to the top event occurrence probability given that the system has failed. Simply, it is the ratio of all MCSs or MCSQs in which an event occurs to the total failure of the system.

Groups and Modules Modularization Technique (GMMT)

A technique for categorizing all dependent events for logical minimization by the use of groups and modules. A module is a set of MCSQs with exactly the same distinct events whilst a group is a set of MCSQs with the same number of distinct events.

Groups and Modules Table (GMT)

A table with groups and modules. Groups could be considered as the columns and modules as the rows.

Inclusive Priority-AND (iPAND)

A logical gate in which all basic events occur one after the other or at the same time. Therefore, for any two input events, *A* and *B*, *A* iPAND *B* means *A* occurs strictly before *B* or *A* and *B* occur at the same time.

Intermediate Event

A combination of events that leads to the occurrence of another event. It is neither a top-event nor a basic event.

Mean Time-to-Failure (MTTF)

The average TTF of a system.

Minimal Cut Set (MCS)

A combination of logically related basic events necessary and sufficient to trigger the occurrence of a top-event. These logically related combinations lack dynamic behaviours; they are usually related by the static gates AND and OR.

Minimal Cut Sequence (MCSQ)

This is analogous to MCS. It is a combination of logically related basic events, with dynamic behaviour, necessary and sufficient to cause a top-event. The dynamic behaviour is modelled with dynamic gates.

Pandora

A temporal fault tree analysis technique that seeks to extend FTA, whilst keeping its simplicity, by the use of novel temporal gates and laws.

parameterized-Simultaneous-AND (pSAND)

A temporal logical gate used to represent the situation where events occur nearly simultaneously – within a relatively short duration of time.

Priority-AND (PAND)

A temporal logical gate representing the sequential occurrence of events; an event occurs then another occurs and on and on.

Priority-OR (POR)

It is a temporal logical gate that represents the occurrence of an event before another or the occurrence of the former event is sufficient to cause the output event.

Reliability

The probability of a system delivering its intended functions, without deviation, under specified conditions within a specific time frame.

Reliability Engineering

A branch of science concerned with the estimation of the reliability of a system. It is part of dependability analysis.

Safety

The measure of a systems ability not to cause devastative effects if it should fail.

Safety-Critical System

A system that will have catastrophic effects on human life and its environment if it should fail. Such systems are also known as high-consequence systems.

Sequential Failure Logic (SFL)

A logic that considers the dynamic/temporal dependencies existing between events. These are inherent in most dynamic safety-critical systems.

Simultaneous-AND (SAND)

It is a temporal logical gate that represents the occurrence of events at exactly the same time. Statistically, nearly impossible for two or exponentially independent events to occur at the same time.

Static Fault Tree (SFT)

A fault tree with only static gates – AND and OR. They are unable to capture the dynamic behaviours.

System

A composition of various logically connected and related functional units performing individual tasks to perform an overall function.

Time-to-Failure (TTF)

The next time to failure of an event.

Temporal Binary Decision Diagram (TBDD)

A binary decision diagram that is able to include temporal features such as sequential failures.

Temporal Fault Tree (TFT)

A fault tree that is capable of capturing the sequential failure present in systems by the use of temporal gates – PAND, SAND, POR. TFTs can be analysed both quantitatively and qualitatively. TFT in this thesis refers to Pandora defined by M. D. Walker not as proposed by G. K. Palshikar.

Top-Event

This corresponds to the total failure of a system under consideration.

Qualitative Analysis

Logical analysis of a fault tree: either static or temporal. Usually involves using logical laws to reduce an expression into its minimal form.

Quantitative Analysis

Probabilistic analysis of a fault tree: static or temporal. Provides the probability of the top-event occurring or importance measures.

Unreliability

The measure of a system not delivering its intended purpose – deviation from intended purpose – under specified conditions it was designed to operate.