Cause-Consequence Analysis of Non-repairable Phased Missions

Gintare Vyzaite, Sarah Dunnett and John Andrews[1]
Department of Aeronautical and Automotive Engineering, Loughborough University, Loughborough, LE11 3TU, UK

Abstract

Many systems can be modelled as a mission made up of a sequence of discrete phases. Each phase has a different requirement for successful completion and mission failure will result if any phase is unsuccessful. Fault tree analysis and Markov techniques have been used previously to model this type of system for non-repairable and repairable systems respectively. Cause-consequence analysis is an alternative assessment technique capable of modelling all system outcomes on one logic diagram. The structure of the diagram has been shown to have advantageous features in both its representation of the system failure logic and its subsequent quantification, which could be applied to phased mission analysis.

This paper outlines the use of the cause-consequence diagram method for systems undergoing non-repairable phased missions. Methods for construction of the cause-consequence diagram are first considered. The disjoint nature of the resulting diagram structure can be utilised in the later quantification process. The similarity with the Binary Decision Diagram method enables the use of efficient and accurate solution routines.

Keywords: cause-consequence diagram, phased mission analysis

Introduction

Many systems perform a mission, which can be divided into consecutive time periods – phases. In each phase, the system needs to perform a specific task. The system configuration, the phase duration, and the failure rates of components often vary from phase to phase. Burdick *et al* [1] describe a phased mission as a task to be performed by a system during executions of which the system is altered such that the logic model changes at specified times. Thus, during a phased mission, time periods (phases) occur in which either the system configuration, system failure characteristics, or both are distinct from those of any immediately succeeding phase.

The most important aim of phased mission analysis is to calculate exact, or obtain bounds for, the mission unreliability. This is defined as the probability that the system fails to function successfully in at least one phase. Estimating the mission reliability by the product of the phase reliabilities results in inaccuracies (in the system reliability), since basic events are shared among logic models of the various phases which are not then independent. Esary and Ziehms [2] used a fault tree method for the analysis of the

[1] Corresponding author: Professor John Andrews, email:J.D.Andrews@lboro.ac.uk

phased missions for non-repairable systems. They introduced basic event transformation and cut set cancellation techniques. But the method proposed by Esary and Ziehms was unable to calculate the probability of failure of each phase due to cut set cancellation, only of the whole mission. La Band and Andrews [3] introduced a new method based on non-coherent fault trees that determines the probability of failure of each phase in addition to the whole mission unreliability. The method combines the causes of success of previous phases with the causes of failure for the phase being considered to allow both qualitative and quantitative analysis of both phase failure and mission failure.

The Binary Decision Diagram (BDD) gives a representation of the system failure logic which is in a format more effective for analysis than that of a fault tree. As such BDDs offer efficient mathematical manipulation, but it is difficult to construct a BDD directly from the system definition. For larger fault trees it is more efficient to convert to a BDD before analysis. Zang, Sun and Trivedi [4] proposed an algorithm for analysis of phased mission systems based on binary decision diagrams. The method determines only the unreliability of the whole mission. La Band and Andrews [3] also use a BDD methodology to evaluate the probability of failure of each phase in the mission.

The cause-consequence diagram method was developed at RISO Laboratories, Denmark, as a graphical tool for analysing relevant accidents in a complex nuclear power plant. The method presents logical connections between causes of an undesired (critical) event and the consequences of such an event, if one or more mitigating provisions fail. As all consequence sequences are investigated, the method can assist in identifying system outcomes, which may not have been investigated at the design stage. Ridley and Andrews [5] notice that, for some types of system, the final cause-consequence diagram has an identical structure to that of the BDD. They noted however that the cause-consequence diagram was more concise due to the automatic extraction of common independent sub-modules. As the cause-consequence diagram can be obtained directly from the system description, there was no need to develop and convert from a fault tree to BDD. They noted that as the BDD is a more efficient tool than the fault tree method then the cause-consequence diagram formulation can be advantageous.

The cause-consequence diagram has the potential to offer a clear representation of phased mission system analysis and also provides an effective quantification technique. The cause-consequence diagram has also the capability to model the failure of each phase in addition to the whole mission in one diagram.

Although the original cause-consequence work developed at RISO was for application to nuclear systems the current methodology is restricted to systems whose components are non-repairable for the duration of the mission. This type of system is typical of those experienced in the aeronautical and aerospace industries. It is also applicable to some military systems under battle conditions.

Cause-Consequence Diagram Method

The cause-consequence diagram method is based on the occurrence of a critical event, which for example may be an event, involving the failure of components or subsystems that is likely to produce hazardous consequences. Once a critical event has been identified, all relevant causes of it and its potential consequences are developed using two conventional reliability analysis methods – fault tree analysis and event tree analysis, Nielsen [6]. Fault tree analysis is used to describe the causes of an undesired event. Event tree analysis shows the consequences that a critical event may lead to if one or more protection systems do not function as designed. The relationship between the two reliability methods is shown in Figure 1.

The main symbol used in the construction of the cause-consequence diagram is a decision box which contains a system condition. It is an identical representation of the 'YES-NO' branches seen on an event tree structure. Following the outlet branches YES and NO of the decision box the diagram is developed. The causes of system failure are developed using fault tree analysis. The cause-consequence diagram terminates in consequence boxes.

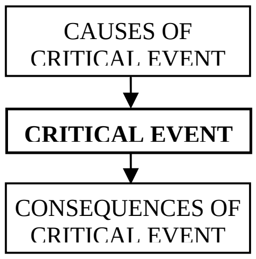

Figure 1. A cause-consequence diagram representation

As an example of the method consider a simple motor control system which is designed to start a motor and then let it run for a limited period of time. The circuit powering the motor is energised by relay contacts closing. The cause-consequence diagram for the system is shown in Figure 2. The first decision box system condition 'Relay contacts open' is considered. If the condition is satisfied and the relay contacts are open then the consequence for the system is that motor stops, this is shown on the YES branch. If the condition is not satisfied and the relay contacts are not open then all other system conditions are considered in turn. In this example 'Timer contacts open' is the next condition considered. In the example the cause of the relay contacts not opening and the fuse not melting are developed using fault trees as shown by the arrows to the first and third decision boxes.

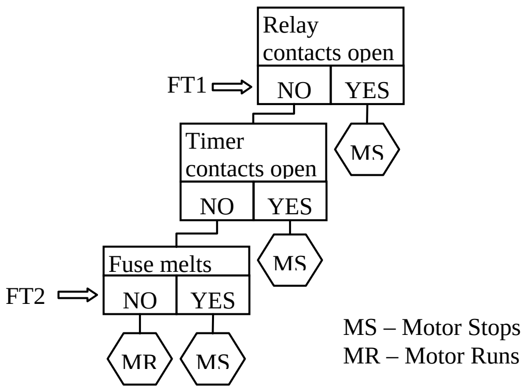

Figure 2. Simple cause-consequence diagram

Cause-Consequence Analysis

The objective of a cause-consequence diagram is to evaluate the likelihood or frequency of each outcome that can result from the critical event. When the probabilities of the decision boxes of the cause-consequence diagram are independent the quantification can be done in following steps:

1. Assign probabilities to each outlet branch stemming from the decision box (possibly by quantifying the relevant fault tree).
2. The probability of any sequence is obtained by multiplying the relevant probabilities associated with each decision box exit path in that sequence.
3. The probability of any consequence is obtained by summing the probabilities of each sequence terminating in that consequence.

However, this procedure cannot be used unless the failures in each decision box in a sequence are independent. Dependencies can exist in cause-consequence diagrams due to repeated or inconsistent events and these must be dealt before quantification.

In the case of repeated events the same failure event exists in more than one fault tree structure on the same path of a cause-consequence diagram. To deal with a common event failure the following algorithm is suggested, Andrews and Ridley [5, 7]:

1. Extract the common event from the fault tree structures and place it in a new decision box preceding the first decision box that contains the common failure event (the new event is the occurrence of the common event: YES/NO).
2. Duplicate the original cause-consequence diagram on each outlet branch stemming from the new decision box.
3. Following the NO outlet branch of the new decision box, the failure event is set to TRUE in any fault tree structure in which it is found.
4. Similarly, following the YES outlet branch, the probability of failure of the common failure event is set to FALSE in any fault tree structure in which it is present.

The other dependency that can arise in the cause-consequence diagram is that the same component may be required to perform different functions which, if successfully accomplished, results in the components residing in different states at different times [5,7]. For example, initially a relay may be required to be closed and later in the sequence to be open. Inconsistent failure events would be that the relay would be failed open to cause the relay to open as required and then failed closed as the cause for the relay to close. The relay cannot be failed open and closed at the same time. Following the identification of inconsistent failure events, the second failure mode in sequence is inspected and, depending on whether the second failure mode is an unrevealed or revealed failure event, the cause-consequence diagram is different.

If the second failure mode is a revealed failure, then it cannot fail between operations and be undetected. Therefore, the time to failure of the second failure mode is set equal to the time it takes the system to operate between the first failure event and the second failure event.

If the second failure mode is unrevealed, then it can occur between operations and be undetected. In this case Ridley and Andrews suggest the following algorithm:

1. Extract either of the inconsistent failure events and place it in a new decision box preceding the first decision box that contains either of the inconsistent events.
2. Duplicate the original cause-consequence diagram on each outlet branch stemming from the new decision box.

3. Following the outlet branch that the extracted component fails all occurrences of the extracted event are set to be TRUE.
4. Following the outlet branch that the extracted component does not fail all occurrences of the extracted event are set to be FALSE.

Following the inspection of each sequence path in the cause-consequence diagram, and modification due to any identified dependent failure events, the cause-consequence diagram can be quantified by following the same procedure as for independent systems. This method does however need to account for the fact that on the path through the diagram which has the working state of the component with the inconsistent failure modes its probability is 1 minus the probability of the inconsistent failure modes.

Phased Mission Analysis Using Cause-Consequence Diagram

To illustrate phased mission analysis using the cause-consequence diagram method, consider the example of the non-repairable system shown in Figure 3.

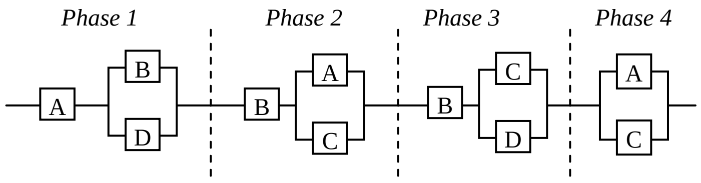

Figure 3. Simple phased mission system

In this example, the system is required to work successfully through all four phases to complete the mission. In phase 1 component A along with B or D are required to work. Failure of component A or components B and D would lead to mission failure. If phase 1 is completed successfully, the system enters phase 2. To complete the mission successfully, the system requirements must be satisfied for each phase.

Considering the phases separately, the fault trees representing individual phase failures are shown in Figure 4.

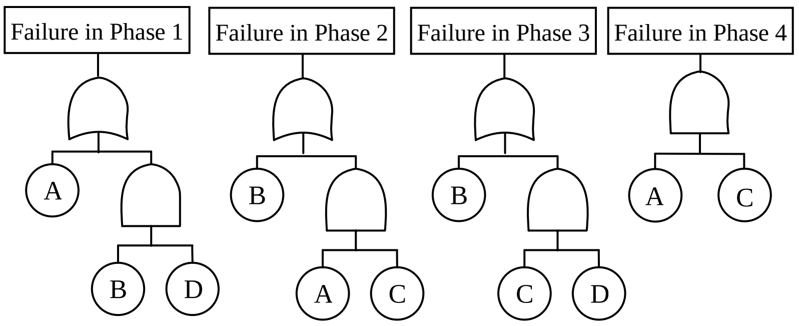

Figure 4. Fault tree representation of individual phase failures

If a cause-consequence analysis is to be performed the challenge is to develop the diagram in an efficient way. The entry point of the diagram is the start of the mission. Consequence events are the failure during each of the phases or mission success.

Cause-Consequence Diagram Construction Methods

Two methods are used to generate the cause-consequence diagram and subsequent analysis performed. The advantages and disadvantages of each method are discussed.

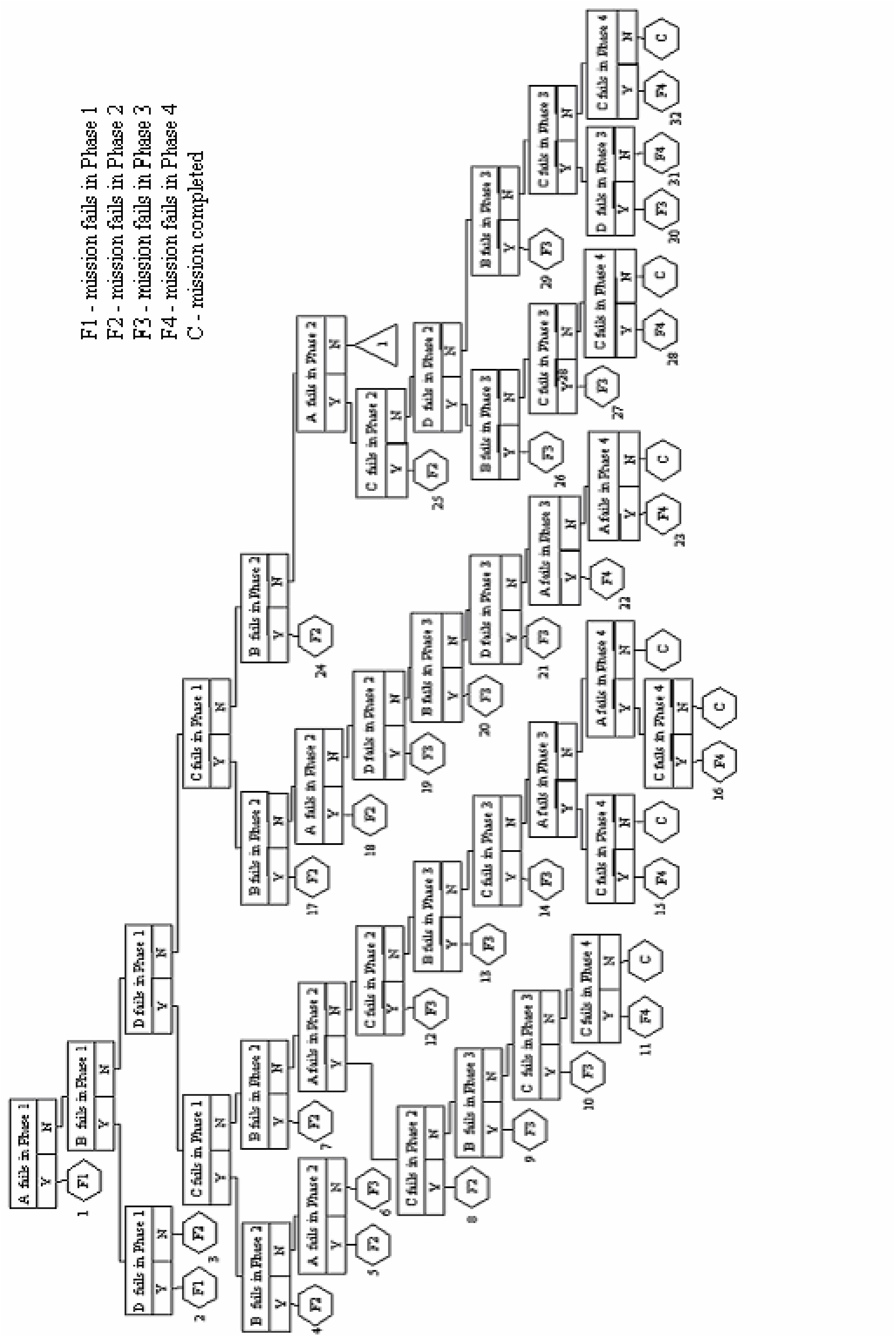
F1 - mission fails in Phase 1
F2 - mission fails in Phase 2
F3 - mission fails in Phase 3
F4 - mission fails in Phase 4
C - mission completed

Figure 5. Cause-consequence diagram for system shown in Figure 3

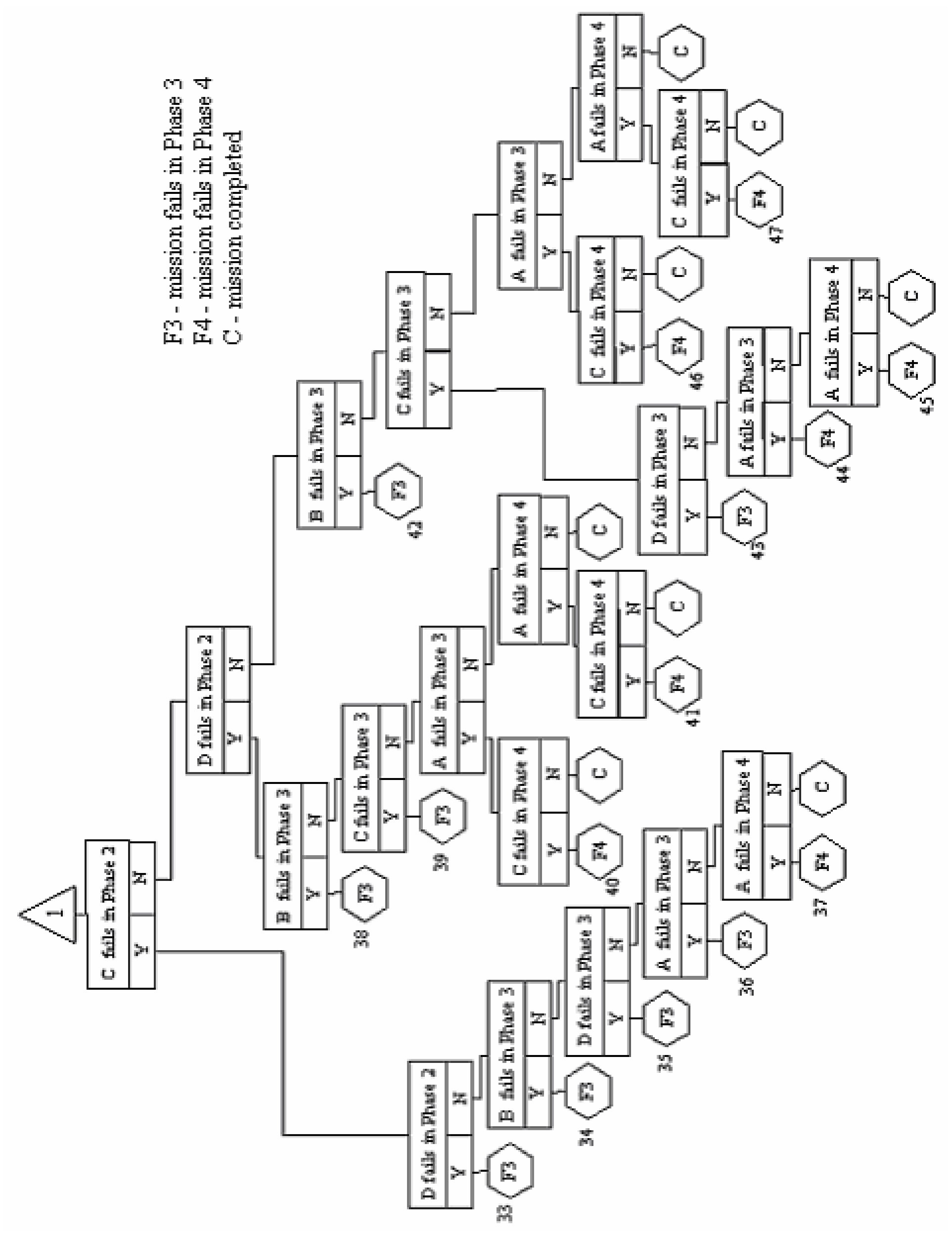

Figure 5. Continued - Cause-consequence diagram for system shown in Figure 3

Method 1

One way of constructing the cause-consequence diagram is to assume a certain order in which the component failure events will be considered. For this example assume the following order:

$$A1 < B1 < D1 < C1 < B2 < A2 < C2 < D2 < B3 < C3 < D3 < A3 < A4 < C4,$$

where A1 means component A fails in phase 1, B2 - component B fails in phase 2, etc. $\overline{A1}$ will mean that component works throughout phase 1.

Following the order of component failures given, each component failure event is added to the diagram as a decision box one by one. That means first consider component A in phase 1: if component A fails in phase 1, then the system state is determined and the system fails in phase 1 and the mission is failed. If component A is functioning throughout phase 1, then consider component B in phase 1: if component B fails, then consider component D in phase 1. If D fails in phase 1, then mission is failed in phase 1. If component D doesn't fail in phase 1, the mission progresses to phase 2, however since B is failed in phase 1 the system fails immediately on entering phase 2. If component B doesn't fail in phase 1, component D is considered. The successful functioning of component D throughout phase 1 following the success of components A and B means the successful mission progress to phase 2. Components are again considered one after the other until the conditions are met for phase 2 failure or progression to phase 3. The complete cause-consequence diagram is shown in Figure 5. In the diagram in Figure 5, consequences F1, F2, F3 and F4 mean mission failure due to system failure in phase 1, 2, 3 and 4 respectively, C represents mission success.

Qualitative Analysis

The qualitative analysis of the cause-consequence diagram will produce the list of causes for each outcome condition. Conditions causing any outcome event are established by investigating each decision box on the path to the outcome and listing the component failure or success in the phase as indicated by the exit path from the decision box. In the example considered here there are 47 failure outcomes, numbered 1 – 47 in figure 5. The component conditions for each of these outcomes was determined, as an example the failure outcomes resulting in mission failure in phases 1 and 2 are listed below.

F1 – Mission failure in Phase 1

1. $A1$
2. $\overline{A1} \wedge B1 \wedge D1$

F2 – Mission failure in Phase 2

3. $\overline{A1} \wedge B1 \wedge \overline{D1}$
4. $\overline{A1} \wedge \overline{B1} \wedge D1 \wedge C1 \wedge B2$
5. $\overline{A1} \wedge \overline{B1} \wedge D1 \wedge C1 \wedge \overline{B2} \wedge A2$
7. $\overline{A1} \wedge \overline{B1} \wedge D1 \wedge \overline{C1} \wedge B2$
8. $\overline{A1} \wedge \overline{B1} \wedge D1 \wedge \overline{C1} \wedge \overline{B2} \wedge A2 \wedge C2$
17. $\overline{A1} \wedge \overline{B1} \wedge \overline{D1} \wedge C1 \wedge B2$
18. $\overline{A1} \wedge \overline{B1} \wedge \overline{D1} \wedge C1 \wedge \overline{B2} \wedge A2$
24. $\overline{A1} \wedge \overline{B1} \wedge \overline{D1} \wedge \overline{C1} \wedge B2$
25. $\overline{A1} \wedge \overline{B1} \wedge \overline{D1} \wedge \overline{C1} \wedge \overline{B2} \wedge A2 \wedge C2$

Each failure mode in the list contains a progression of states for the same component. For example, outcome 4 has component B working throughout phase 1 and then failing in phase 2. The list can therefore be simplified.

When there is a situation in the list where a component is working in an earlier phase, but fails in a later phase, the event that component was working in earlier phase can be cancelled out as failure in later phase implies that it was working before. This is the situation in several outcomes. The simplified list of mission failures in phase 1 and 2 is listed below:

F1 – Mission failure in Phase 1

1. $A1$
2. $\overline{A1} \wedge B1 \wedge D1$

F2 – Mission failure in Phase 2

3. $\overline{A1} \wedge B1 \wedge \overline{D1}$
4. $\overline{A1} \wedge D1 \wedge C1 \wedge B2$
5. $\overline{B1} \wedge D1 \wedge C1 \wedge \overline{B2} \wedge A2$
7. $\overline{A1} \wedge D1 \wedge \overline{C1} \wedge B2$
8. $\overline{B1} \wedge D1 \wedge \overline{B2} \wedge A2 \wedge C2$
17. $\overline{A1} \wedge \overline{D1} \wedge C1 \wedge B2$
18. $\overline{B1} \wedge \overline{D1} \wedge C1 \wedge \overline{B2} \wedge A2$
24. $\overline{A1} \wedge \overline{D1} \wedge \overline{C1} \wedge B2$
25. $\overline{B1} \wedge \overline{D1} \wedge \overline{B2} \wedge A2 \wedge C2$

Quantitative Analysis

The reduced or simplified lists of component conditions leading to each outcome are in an appropriate form for quantification. This is because each of the outcome event sequences are mutually disjoint. Under these conditions the probability of achieving any particular phase failure outcome is the sum of the probabilities leading to that outcome.

Quantification of the diagram starts with the calculation of the probabilities for each event X failing in phase i having worked throughout the previous phases, $\mathbf{P}(X_i)$. The probabilities are usually evaluated over the relevant phase period by integrating the component failure density function $f_X(t)$. So if phase i is from t_{i-1} to t_i then

$$\mathbf{P}(X_i) = \int_{t_{i-1}}^{t_i} f_X(t)dt \tag{1}$$

When component is working in two (or more) consecutive phases, for example 'component A works through phase 1' and 'component A works through phase 2', probability is calculated as:

$$\mathbf{P}\left(\overline{A1} \wedge \overline{A2}\right) = 1 - \mathbf{P}(A1) - \mathbf{P}(A2)$$

So, the probability of mission failure for this system would be:

$$\mathbf{P}(mission \quad failure) = \mathbf{P}(F1) + \mathbf{P}(F2) + \mathbf{P}(F3) + \mathbf{P}(F4),$$

where

$$\begin{aligned}
\mathbf{P}(F1) &= \mathbf{P}(A1) + \mathbf{P}\left(\overline{A1}\right)\mathbf{P}(B1)\mathbf{P}(D1) \\
\mathbf{P}(F2) &= \mathbf{P}\left(\overline{A1}\right)\mathbf{P}(B1)\mathbf{P}\left(\overline{D1}\right) + \mathbf{P}\left(\overline{A1}\right)\mathbf{P}(D1)\mathbf{P}(C1)\mathbf{P}(B2) + \\
&\quad + (\mathbf{1} - \mathbf{P}(B1) - \mathbf{P}(B2))\mathbf{P}(D1)\mathbf{P}(C1)\mathbf{P}(A2) + \mathbf{P}\left(\overline{A1}\right)\mathbf{P}(D1)\mathbf{P}\left(\overline{C1}\right)\mathbf{P}(B2) + \\
&\quad + (\mathbf{1} - \mathbf{P}(B1) - \mathbf{P}(B2))\mathbf{P}(D1)\mathbf{P}(A2)\mathbf{P}(C2) + \mathbf{P}\left(\overline{A1}\right)\mathbf{P}\left(\overline{D1}\right)\mathbf{P}(C1)\mathbf{P}(B2) + \\
&\quad + (\mathbf{1} - \mathbf{P}(B1) - \mathbf{P}(B2))\mathbf{P}\left(\overline{D1}\right)\mathbf{P}(C1)\mathbf{P}(A2) + \mathbf{P}\left(\overline{A1}\right)\mathbf{P}\left(\overline{D1}\right)\mathbf{P}\left(\overline{C1}\right)\mathbf{P}(B2) + \\
&\quad + (\mathbf{1} - \mathbf{P}(B1) - \mathbf{P}(B2))\mathbf{P}\left(\overline{D1}\right)\mathbf{P}(A2)\mathbf{P}(C2)
\end{aligned}$$

Expressions were also determined for $\mathbf{P}(F3)$ and $\mathbf{P}(F4)$ and then the probability of mission failure determined.

The result obtained was found to be the same as that using the fault tree method and BDD analysis proposed by La Band and Andrews [3].

Method 2

It would appear that the quantification of the cause-consequence analysis diagram for phased mission is efficient and straight forward. The difficulty will come in obtaining the cause-consequence diagram in the first place. For the simple system shown in Figure 4 there was no problem. But as the size and complexity of the system increases so does the cause-consequence diagram and it requires an alternative method which could be automated on a computer to make this a practical proposition.

The alternative method is based on the fact that the whole cause-consequence diagram contains events which are both repeated and inconsistent for each component.

The first step of the second method of cause-consequence diagram construction is shown in Figure 6 with corresponding fault trees shown in Figure 8. The diagram is at this stage at a high level of abstraction and just considers phase failure. The first decision box contains the failure event 'Phase 1 fails', if it is true, then mission fails in phase 1 (failure F1). If the system works successfully throughout phase 1, it progresses to phase 2. If the system fails in phase 2, then the cause-consequence diagram terminates with failure event 'Mission fails in phase 2' (F2). This is repeated for phases 3 and 4. If the system doesn't fail in phase 4, then the mission is completed successfully. Phase failure causes are developed using fault tree analysis and the relevant fault trees are attached to the YES branches of the decision box.

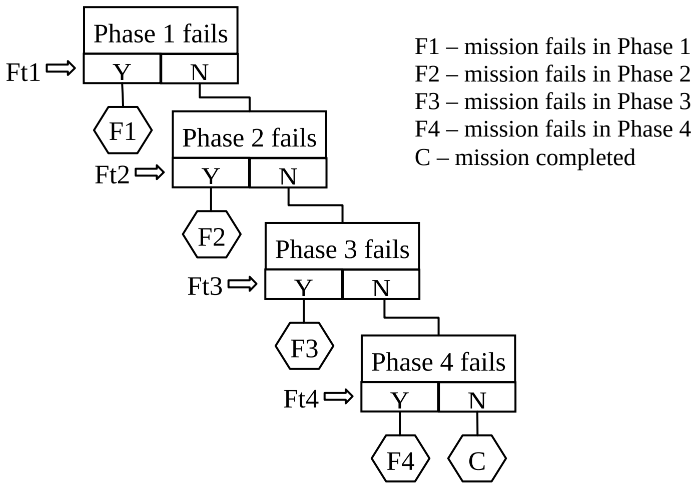
Phase 1 fails
Ft1
Y
N
F1
Phase 2 fails
Ft2
Y
N
F2
Phase 3 fails
Ft3
Y
N
F3
Phase 4 fails
Ft4
Y
N
F4
C
F1 – mission fails in Phase 1
F2 – mission fails in Phase 2
F3 – mission fails in Phase 3
F4 – mission fails in Phase 4
C – mission completed

Figure 6. Construction of cause-consequence diagram – step 1

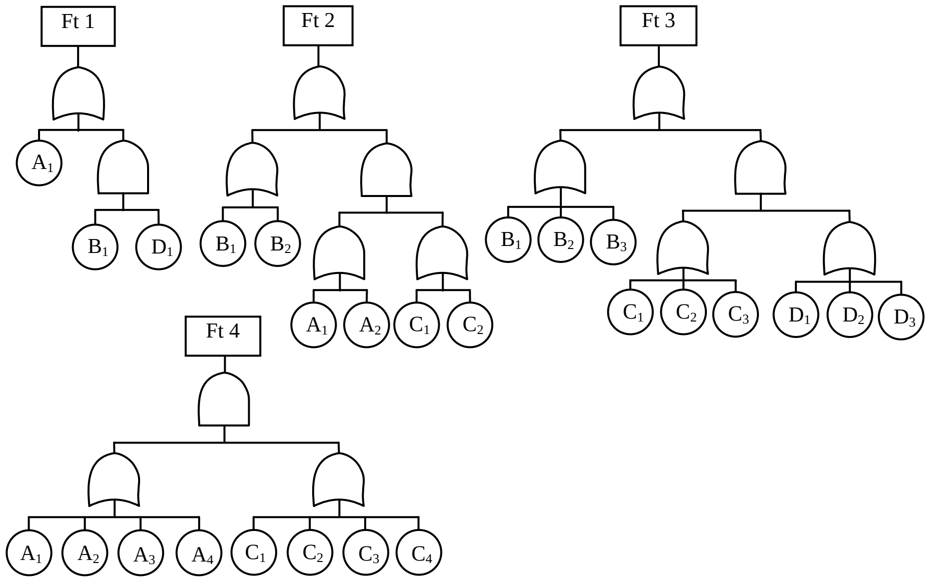

Figure 7. Fault trees for the cause-consequence diagram shown in Figure 6

The fault trees in Figure 7 are obtained using the basic event transformation introduced by Esary and Ziehms [2]. They noted, that, since the system is non-repairable, a component functions in phase i if, and only if, it has previously functioned in phase 1, and in phase 2, …, and in phase $i-1$, and then functions in phase i. So, component failure events in each phase fault tree are replaced by an OR combination of the failure events of that and all preceding phases. For example, the condition that component A is in the failed state in phase 2 would be replaced by the OR gate with the failure of component A in phase 1 ($A1$), and the failure of component A in phase 2 ($A2$) as inputs.

As the cause-consequence diagram contains both repeated and inconsistent failure events, these must be extracted one by one following the normal cause-consequence analysis procedure described earlier. An order in which the component failure events are to be considered for extraction. For this example we assume a different order of component failure events than for Method 1:

$$A1 < B1 < B2 < A2 < C1 < C2 < B3 < C3 < D1 < D2 < D3 < A3 < A4 < C4 .$$

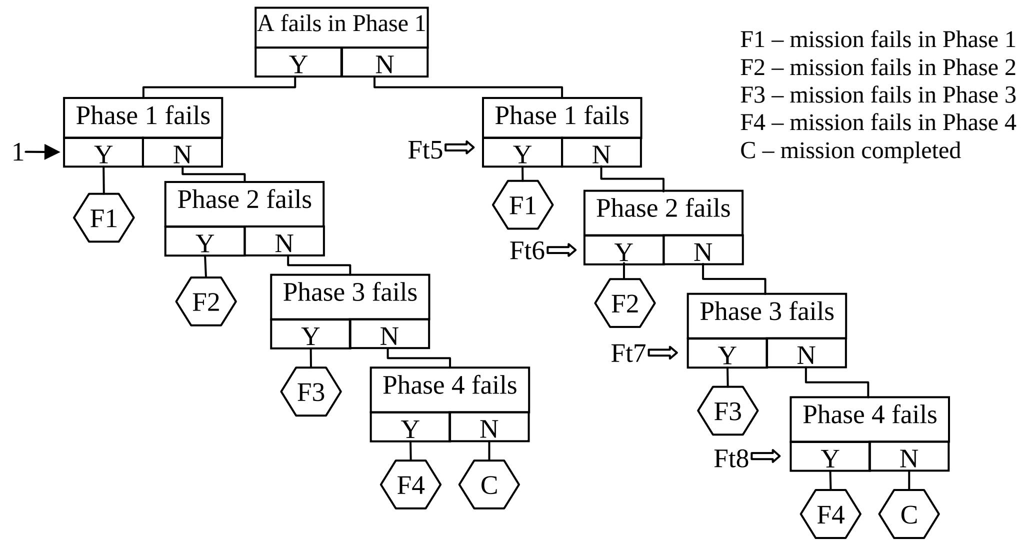

Figure 8. Construction of cause-consequence diagram – step 2

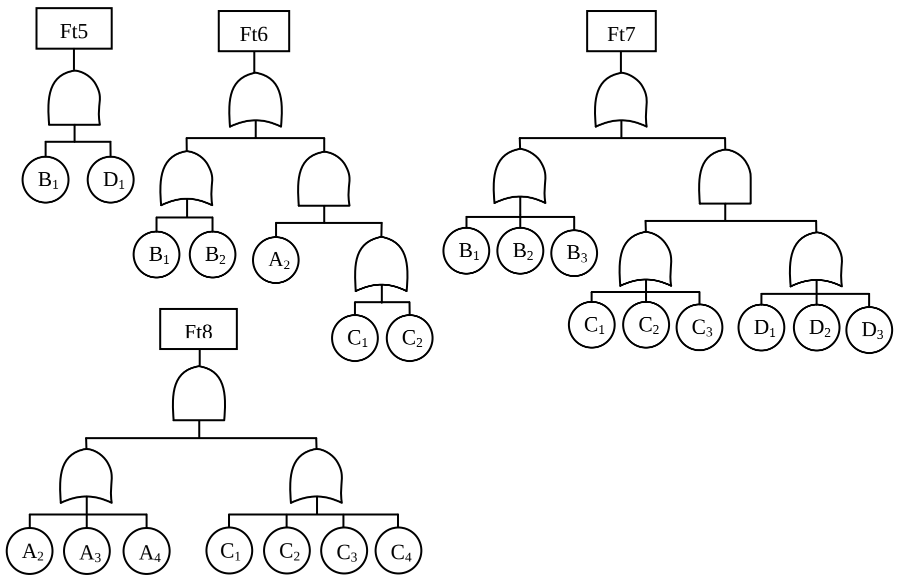

Figure 9. Fault trees for the cause-consequence diagram shown in Figure 8

Then following the normal cause-consequence analysis process the event ‘A fails in phase 1’ (A1), as it is both a repeated and an inconsistent event, can be extracted from fault tree structures and placed in a new decision box preceding the first decision box that contains it (‘Phase 1 fails’). Then the diagram is duplicated on both outlet branches and following the YES branch all occurrences of the event are set to TRUE, and following NO branch all occurrences of this event are set to be FALSE. The cause-consequence diagram for this step is shown in Figure 8 and the fault trees in Figure 9.

As the probability of event ‘Phase 1 fails’ on the YES branch of the decision box ‘A fails in phase 1’ is 1, the diagram can be reduced (Figure 10).

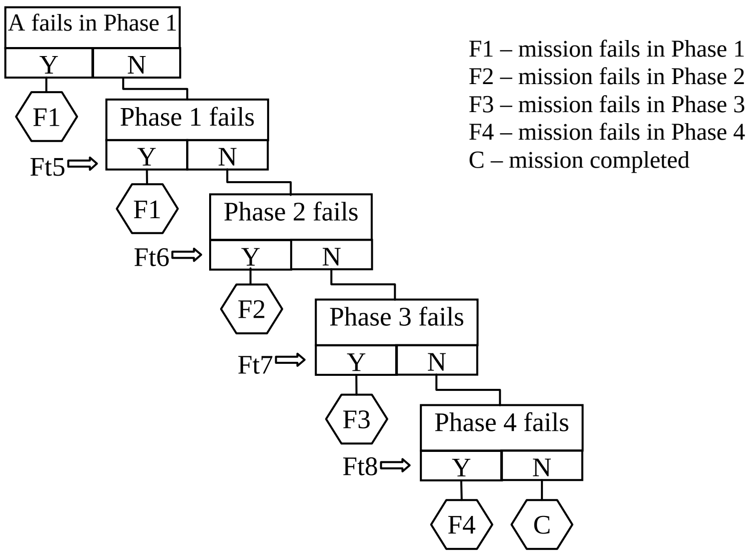

Figure 10. Construction of cause-consequence diagram – step 2 (reduced)

Following the order of component failure events the process is repeated and events are extracted from fault tree structures and placed in new decision boxes. The diagram is developed using normal cause-consequence analysis process. The final cause-consequence diagram obtained following this procedure is shown in Figure 11.

Once the cause-consequence diagram is obtained the same procedures for qualitative and quantitative analysis as for Method 1 should be followed. The cause-consequence diagram obtained using Method 2 gives the same result for the mission failure probability as the one obtained using Method 1.

From these two different methods it can be noticed that component ordering is important, as different variable orderings will produce different size diagrams. Both methods will give the same quantitative results, but they might produce different cause-consequence diagrams. For example, using a different order of component failure events in Method 2 gives only 32 system outcome events compared with 47 in Method 1.

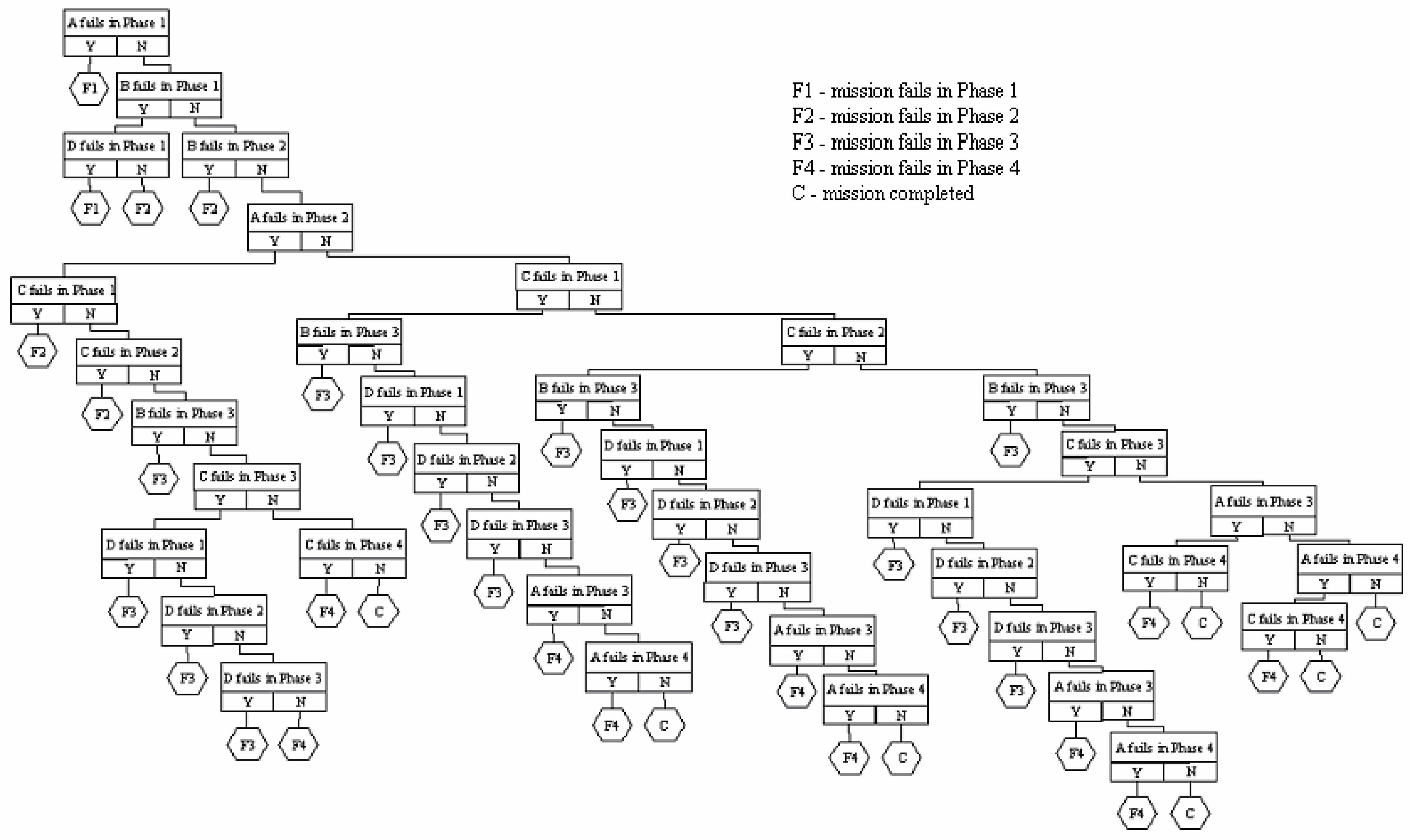

Figure 11. Final cause-consequence diagram

Discussion

It has been shown here, from the example considered, that different ordering of component failure events can significantly reduce the size of the cause-consequence diagram. The selection of an effective variable ordering scheme can produce a very efficient cause-consequence diagram for the phased mission problem. Method 2 is easier to automate and hence easier to implement as a computer code for extraction.

The methods shown above offer different ways of constructing the cause-consequence diagram. The quantification procedure is the same for both methods. Summarizing, the procedure is as follows:

1. Construct the cause-consequence diagram using one of above methods.
2. Write out the list of component conditions to produce each outcome
3. If there is a situation where a component is working in earlier phases but fails in later phase, take out working states of that component.
4. Calculate the probability of each outcome. When a component is working in two (or more) consecutive phases, for example 'component A works through phase 1' and 'component A works through phase 2', probability is calculated as:

$$\mathbf{P}\left(\overline{A1} \wedge \overline{A2}\right) = 1 - \mathbf{P}(A1) - \mathbf{P}(A2)$$

5. To obtain the probability of a certain consequence, sum the probabilities of all outcomes ending in that consequence.

Conclusions

An algorithm of constructing and analysing a cause-consequence diagram to solve a non-repairable phased mission problem has been developed. The method gives an exact probability of mission failure and of each phase. The cause-consequence diagram can be constructed directly from the fault trees representing the causes of each phase failure.
As the final cause-consequence diagram has no fault trees attached to any decision boxes, it has the same form as a BDD. This has the advantage over other methods developed which use BDDs for phased mission analysis. In these each phase has to be modelled separately. The proposed cause-consequence method can model all phases in one diagram resulting in a more efficient analysis.

References

1. Burdick G.R., Fussell J.B., Rasmuson D.M., Wilson J.R., The Implementation of Phased Mission Techniques To Nuclear Systems Analysis, in *A Collection of Methods for Reliability and Safety Engineering*, Report IV, Idaho National Engineering Laboratory, ANCR-1273 pp 89-119 (1976)
2. Esary J.D., Ziehms H., Reliability Analysis of Phased Missions, in *Reliability and Fault Tree Analysis: Theoretical and Applied Aspects on System*, Reliability and Safety assessment, Philadelphia, pp 213-236 (1975)
3. La Band R.A., Andrews J.D., Phased Mission Modelling Using Fault Tree Analysis, *Proc Instn Mech Engrs Part E: Journal of Process Mechanical Engineering*, **218**[2] 83-91 (2004)
4. Zang X., Sun H., Trivedi K.S., A BDD-Based Algorithm for Reliability Analysis of Phased-Mission Systems, in *IEEE Transactions on Reliability*, **48**[1] 50-60 (1999)

5. Andrews J.D., Ridley L.M., Application Of Cause-Consequence Diagram Method To Static Systems, in *Reliability Engineering and System Safety*, **75**[1] 47-58 (2002)
6. Nielsen D.S., The Cause/consequence Diagram Method as a Basis for Quantitative Accident Analysis, *Danish Atomic Energy Commission,* RISO-M-1374, May 1971
7. Andrews J.D., Ridley L.M., Reliability of Sequential Systems Using the Cause-Consequence Method, *Proc Instn Mech Engrs Part E: Journal of Process Mechanical Engineering*, **215**[3] 207-220 (2001)