# A conservative confidence bound for the probability of failure on demand of a software-based system based on failure-free tests of its components

Peter Bishop, Andrey Povyakalo

*City, University of London*

**Abstract**

The standard approach to deriving the confidence bound for the probability of failure on demand (*pfd*) of a software-based system is to perform statistical tests on the whole system as a "black-box". In practice, performing tests on the entire system may be infeasible for logistical reasons, such as lack of availability of all component subsystems at the same time during implementation. This paper presents a general method for deriving a confidence bound for the overall system from successful independent tests on individual system components. In addition, a strategy is presented for optimizing the number of tests allocated to system components for an arbitrary system architecture that minimizes the confidence bound for the system *pfd*. For some system architectures, we show that an optimum allocation of component tests is as effective as tests on the complete system for demonstrating a given confidence bound.

The confidence bound calculation makes use of many of the concepts used in the reliability analysis of hardware structures, but unlike a conventional hardware analysis, the method does not presume statistical independence of failures between software components, so the confidence bound calculation for the software should always be conservative.

*Keywords:* Statistical testing, Confidence bounds, Software reliability, Fault tolerance

## 1. Introduction

The standard approach to deriving a confidence bound for a software-based system is to perform statistical tests on the system as a "black-box" [1]. In practice, however, it may be impractical to test the entire system for logistical reasons. For example, the hardware for only one component can be made available for statistical test purposes – this was a constraint in the statistical reliability testing of the computerized protection systems at both Sizewell B [2] and Hinkley Point C [3] nuclear power stations. In this paper we assume that independent statistical tests can be applied to the individual software-based components that form part of a redundant complex system.

We first describe the general approach to statistical testing and outline the technical challenges addressed by our paper. Next we summarize the confidence bound theory for a system tested as a "black-box" system and then extend the theory to derive a conservative, analytic confidence bound for a system based on the number of component tests and the structural redundancy of the system. We also identify two sub-optimal distributions of component tests that can improve the system confidence bound.

Using this generic model, we derive specific confidence bounds for different system structures and optimized component test plans for each structure. The approach is illustrated using numerical examples.

## 2. Background to Statistical Testing

For UK nuclear safety systems, the Office for Nuclear Regulation (ONR) requires a safety case to demonstrate "production excellence" combined with "independent confidence building measures" [4]. The demonstration of production excellence includes a requirement to meet all relevant standards for nuclear safety systems such as IEC 61226 [5] and IEC 61513 [6] for the hardware and IEC 60880 [7] for the software. Production excellence should ensure the supplied system has the required safety integrity and reliability attributes, but a further independent confidence building stage is required for the final delivered system.

Statistical testing [8, 9, 1] is an independent confidence building measure that provides an estimate for the software probability of failure on demand (*pfd*) of a demand-based system to some confidence bound, and it is recommended in functional safety standards such as IEC 61508 [10].

Statistical reliability testing was performed as part of the independent confidence building programme required by ONR for the computer-based primary protection system (PPS) at Sizewell B nuclear power station [1]. The final version of the PPS was subjected to 5000 simulated demands to support a *pfd* claim of $10^{-3}$ to the 99% confidence level recommended by ONR [11].

Reliability testing of safety systems is also under way for new nuclear power stations being built in the UK [3] where the target *pfd* could be even more demanding.

The confidence bound derived from statistical reliability testing is based on a number of modelling assumptions. The stated assumptions in IEC 61508 [10] for the low demand rate case (such as a protection system) are:

1. The test data distribution is equal to the distribution of demands during on-line operation.
2. Test runs are statistically independent from each other, with respect to the cause of a failure.
3. An adequate mechanism exists to detect any failures which may occur.
4. Number of test cases $N > 100$.
5. No failure occurs during the $N$ test cases.

To satisfy the first assumption, the test demands should be representative of those encountered in operation. This requires identification of the different events that could trigger a demand, the event frequency and the plant transients that result from different events. For example, in a nuclear plant typical failure events are: loss of reactor coolant, pump failure and control rod failure. Each type of event results in a different plant transient that affects different plant parameters (like temperature, pressure, coolant flows and neutron flux).

The second assumption can be met in the protection system context as the protection system is normally reset after a reactor trip (so the software always starts from the same initial state). Provided the response to a demand is independent of the time between reset and demand, accelerated testing is possible, so the required number of tests can be performed in a realistic period of time.

The third assumption requires a perfect "oracle" that determines if a failure has occurred. The required response is relatively easy to determine in a nuclear plant since each simulated plant transient is expected to result in a reactor trip.

The last two assumptions will also be met in a nuclear protection context as many thousands of tests are needed for the required *pfd* and, in the exceptional event that a failure is observed, the software will be corrected and retested from scratch.

## 3. Contribution of this Paper

Our paper presents a method for deriving a confidence bound for the *pfd* of a software-based system based on failure-free tests performed on its component parts. The method makes use of standard reliability modelling concepts utilized in hardware PRA [12] such as structure functions, cutsets, minimal cutsets, reliability block diagrams (RBD) and success paths, but uses an entirely different approach to quantifying the contribution of the component software to the system *pfd*.

A different quantification approach was needed for the following reasons:

1. *Failure-free tests.* Hardware reliability models are typically based on comprehensive prior failure data of mass-produced components [13]. With multiple failures, the failure probability of a component can be estimated with a known precision, and used within a structural reliability model to derive a "best estimate" for the system *pfd*.
   By contrast, zero failures are observed in a successful statistical software test campaign, so the maximum likelihood estimates for all the failure probabilities are zero. However it is possible to state classic confidence intervals for the actual unknown failure probabilities [14].
2. *Systematic failure.* Unlike hardware, software failures are deterministic. Identical sequences of input values will always cause a software component to fail. Software reliability is determined by the probability that a defective region of the input space is hit during normal operation [15].
   It has been shown empirically [16] and explained theoretically [17, 18] that similar regions in the input space are likely to be defective in different software implementations (e.g. due to a specification flaw).

To address these issues, we use a radically different method for deriving the system *pfd* from the component tests. Since accurate software component failure probabilities are impossible to derive, and dependencies between the component failures are unknown, we instead assign a worst case set of probabilities to the system cutsets that maximizes the system *pfd* confidence bound. Our analysis shows that the bound is maximized when all the failure probability is assigned to a single minimal cutset.

As a result, the model presented in this paper produces results that differ from conventional hardware reliability modelling. This is not because either approach is flawed, it simply reflects differences in the available failure data and the nature of hardware and software failure. In practice, both hardware and software model estimates are needed to compute the overall *pfd* of a computer-based protection system.

## 4. Conventional Confidence Bound Theory

If a system is subjected to $N$ test demands without failure, the $(1-\alpha)$ confidence interval for the system *pfd* includes all the values, which are not rejected by the likelihood ratio (LR) test against the maximum likelihood alternative (H1: $pfd = 0$), at the significance level $\alpha$ [14].

If $pfd = q$, then likelihood of observing no failures in $N$ tests is $(1-q)^N$. The maximum likelihood of this observation (for $q = 0$) is 1, resulting in the likelihood ratio $LR = (1-q)^N$.

Therefore, the $(1-\alpha)$ confidence interval for the unknown value of the system *pfd* is described by the following inequality:

$$LR = (1-q)^N \geq \alpha. \quad (1)$$

with the upper confidence bound:

$$q_s = 1 - \alpha^{1/N}. \quad (2)$$

This bound $q_s$ can be conservatively approximated [8] by a bound $q_u > q_s$ where

$$q_u = \frac{1}{N} \ln(1/\alpha). \quad (3)$$

This confidence bound is established by testing the entire system as a “black-box”. In the next section we present an extension to the standard approach that allows a system confidence bound to be derived by performing statistical tests on the individual components.

| Component | $c_1$ | $c_2$ | $c_3$ | |
|---|---|---|---|---|
| State | $x_1$ | $x_2$ | $x_3$ | $\varphi(\mathbf{x})$ |
| $\mathbf{x}_0$ | 0 | 0 | 0 | 0 |
| $\mathbf{x}_1$ | 1 | 0 | 0 | 0 |
| $\mathbf{x}_2$ | 0 | 1 | 0 | 0 |
| $\mathbf{x}_3$ | 0 | 0 | 1 | 0 |
| $\mathbf{x}_4$ | 1 | 1 | 0 | 1 |
| $\mathbf{x}_5$ | 1 | 0 | 1 | 1 |
| $\mathbf{x}_6$ | 0 | 1 | 1 | 1 |
| $\mathbf{x}_7$ | 1 | 1 | 1 | 1 |

Table 1: Example 2oo3 vote structure function

## 5. Confidence Bounds from Component Tests

Failure-free testing over $m$ individual components can be characterized by a test plan vector:

$$\mathbf{n} = (n_1, n_2, \ldots, n_m)' \quad (4)$$

where $m$ is a number of components, $n_i$ is the number of (failure-free) tests for component $i$, and the total number of tests is:

$$N = \sum_{i=1}^{m} n_i. \quad (5)$$

To determine the attainable confidence upper bound for the system *pfd*, we need to take account of the fact that a failure at the component level does not necessarily result in a failure at the system level if the overall system is designed to be fault tolerant. In the following subsections we consider how the fault tolerant behaviour is defined, and how this can be used to infer a confidence bound for the system based on component tests.

### *5.1. Characterizing Fault Tolerance*

To characterize the fault tolerance capability of the overall system, we first define $\mathbf{x} = (x_1, x_2, \ldots, x_m)'$ as a random binary vector of indicators of component failure. If component $i$ fails, $x_i = 1$ and $x_i = 0$ otherwise.

The failure-proneness of the overall system is represented by a structure function $\varphi(\mathbf{x})$ [19], where $\varphi(\mathbf{x}) = 1$ if the system fails for a given combination of component failures and successes $\mathbf{x}$ (a system state).

We presume $\varphi(\mathbf{x})$ is monotonic, and

$$\varphi(\mathbf{0}) = \varphi(0, 0, \ldots, 0) = 0;$$
$$\varphi(\mathbf{1}) = \varphi(1, 1, \ldots, 1) = 1.$$

For example, Table 1 shows the states for a 2 out of 3 (2oo3) vote structure where two or more component failures will result in system failure (i.e. where $\varphi(\mathbf{x}) = 1$).

This example presumes that all failure states defined by $\varphi(\cdot)$ are determined solely by the interconnection of components within the structure. However, if there are identical software-based components within the structure that are subject to the same inputs, some failure states represented by an architecture-based structure function will be infeasible.

This can be represented formally by redefining the structure function as:

$$\varphi(\mathbf{x}) = \varphi_A(\mathbf{x})\phi_S(\mathbf{x}), \quad (6)$$

where $\varphi_A(\mathbf{x})$ is based solely on the component connection architecture and $\phi_S(\mathbf{x})$ is a further binary indicator function that takes account of software dependencies between components. The $\phi_S(\mathbf{x})$ indicator function can be viewed as an additional filter that excludes system states that are infeasible, i.e. $\phi_S(\mathbf{x}) = 0$ if the combination is infeasible, and $\phi_S(\mathbf{x}) = 1$ if the combination is feasible.

| State | $x_1$ | $x_2$ | $x_3$ | $\varphi_A(\mathbf{x})$ | $\phi_S(\mathbf{x})$ | $\varphi(\mathbf{x})$ |
|---|---|---|---|---|---|---|
| $\mathbf{x}_0$ | 0 | 0 | 0 | 0 | 1 | 0 |
| $\mathbf{x}_1$ | 1 | 0 | 0 | 0 | 0 | 0 |
| $\mathbf{x}_2$ | 0 | 1 | 0 | 0 | 0 | 0 |
| $\mathbf{x}_3$ | 0 | 0 | 1 | 0 | 1 | 0 |
| $\mathbf{x}_4$ | 1 | 1 | 0 | 1 | 1 | 1 |
| $\mathbf{x}_5$ | 1 | 0 | 1 | 1 | 0 | 0 |
| $\mathbf{x}_6$ | 0 | 1 | 1 | 1 | 0 | 0 |
| $\mathbf{x}_7$ | 1 | 1 | 1 | 1 | 1 | 1 |

Table 2: A 2oo3 vote structure function with failure dependency

For cases where all components differ, all combinations are feasible, so $\phi_S(\mathbf{x}) = 1$ for all system states and $\varphi(\mathbf{x})$ is the same as the original structure function $\varphi_A(\mathbf{x})$. On the other hand, Table 2 includes an example filter for the case where components $c_1$ and $c_2$ are identical so their binary failure states must be identical to be feasible. As a result, the only feasible failure states of the system are $\mathbf{x}_4$ and $\mathbf{x}_7$.

### *5.2. Derivation of the System Confidence Bound*

In the following analysis, we will use the structure function $\varphi(\mathbf{x})$ to derive a worst case confidence bound at the system level due to component software failure[1] based on the successful tests of the individual components.

### 5.2.1. Basic Concept

Unlike a PRA calculation, there is *no assumption that the software components fail independently*. For example, component software failures might overlap partly or, in the worst case, completely – as illustrated in Figure 1 where the same input value can trigger a failure in components $c_1$ and $c_2$.

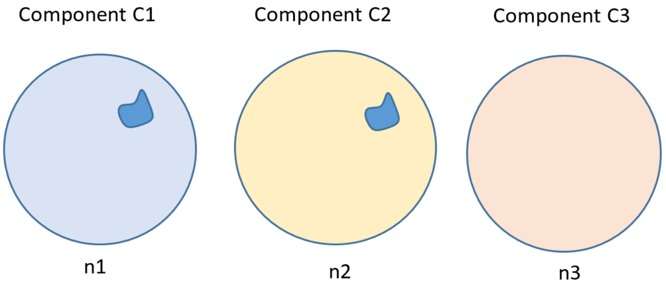

Figure 1: Worst case 2oo3 fault distribution (diverse components)

In practice, we do not know the probability of joint component failures. A conventional PRA might assume independent failures between components to compute the joint failure probabilities. Our approach is more conservative since it identifies the worst case scenario consistent with the system confidence bound. These scenarios are cases where:

- component failures only occur in system states that cause system failure (e.g. concurrent component failures),
- the probability of being in each failure state (cutset) is chosen to maximize the *pfd* confidence bound for the overall system for the given set of tests,
- the assigned probability is always located in a minimal cutset failure state.

While this approach may appear to be overly-pessimistic, it does represent credible extremes like the one shown in Figure 1, where a common fault is present in two software components and only tests on these two components are relevant for deriving the system confidence bound.

### 5.2.2. Structure-based Confidence Bound Model

For a given distribution of system state probabilities $\mathbf{p}(\cdot)$, the *pfd* of the overall system, $Q$ is:

$$Q = \sum_{\mathbf{x}\in\mathbf{B}^m} \varphi(\mathbf{x})p(\mathbf{x}), \tag{7}$$

where $\mathbf{B} = \{0, 1\}$ is a binary vector representing the possible failure states of a single component.

For an individual component $i$, the failure probability $q_i$ will be the sum of all failure state (cutset) probabilities that include a failure of component $i$, i.e.

$$q_i = \sum_{\mathbf{x}\in\mathbf{B}^m} x_i p(\mathbf{x}). \tag{8}$$

For example, if we multiply the 2oo3 structure function in Table 1 by $\mathbf{p}(\cdot)$ we obtain the structure-based probability table shown in Table 3. It can be seen that the failure probability of component $c_1$ is:

$q_1 = p(\mathbf{x}_1) + p(\mathbf{x}_4) + p(\mathbf{x}_5) + p(\mathbf{x}_7)$,

while the system *pfd* is:

$Q = p(\mathbf{x}_4) + p(\mathbf{x}_5) + p(\mathbf{x}_6) + p(\mathbf{x}_7)$.

| $\mathbf{p}(\cdot)$ | $x_1p(\mathbf{x})$ | $x_2p(\mathbf{x})$ | $x_3p(\mathbf{x})$ | $\varphi(\mathbf{x})p(\mathbf{x})$ |
|---|---|---|---|---|
| $p(\mathbf{x}_0)$ | 0 | 0 | 0 | 0 |
| $p(\mathbf{x}_1)$ | $p(\mathbf{x}_1)$ | 0 | 0 | 0 |
| $p(\mathbf{x}_2)$ | 0 | $p(\mathbf{x}_2)$ | 0 | 0 |
| $p(\mathbf{x}_3)$ | 0 | 0 | $p(\mathbf{x}_3)$ | 0 |
| $p(\mathbf{x}_4)$ | $p(\mathbf{x}_4)$ | $p(\mathbf{x}_4)$ | 0 | $p(\mathbf{x}_4)$ |
| $p(\mathbf{x}_5)$ | $p(\mathbf{x}_5)$ | 0 | $p(\mathbf{x}_5)$ | $p(\mathbf{x}_5)$ |
| $p(\mathbf{x}_6)$ | 0 | $p(\mathbf{x}_6)$ | $p(\mathbf{x}_6)$ | $p(\mathbf{x}_6)$ |
| $p(\mathbf{x}_7)$ | $p(\mathbf{x}_7)$ | $p(\mathbf{x}_7)$ | $p(\mathbf{x}_7)$ | $p(\mathbf{x}_7)$ |
| 1 | $q_1$ | $q_2$ | $q_3$ | $Q$ |

Table 3: 2oo3 structure probability table

Given component failure probabilities $\mathbf{q} = (q_1, \ldots q_m)'$, the likelihood of observing zero failures in $m$ components with test plan $\mathbf{n}$ is:

$$L(\mathbf{q}, \mathbf{n}) = \prod_{i=1}^{m} (1 - q_i)^{n_i}. \tag{9}$$

This is equivalent to equation (1) for the case where there is only a single black-box component i.e. $m = 1$.

Re-expressing (9) in terms of system state probabilities, the likelihood function given the observation is:

$$L(\mathbf{p}(\cdot), \mathbf{n}) = \prod_{i=1}^{m} \left(1 - \sum_{\mathbf{x}\in\mathbf{B}^m} x_i p(\mathbf{x})\right)^{n_i} \tag{10}$$

$L(\mathbf{p}(\cdot), \mathbf{n})$ reaches its maximum value 1, when $p(\mathbf{0}) = 1$ and $p(\mathbf{x}) = 0$ for all other $\mathbf{x} \neq \mathbf{0}$, resulting in the likelihood ratio against the maximum likelihood alternative equal to $LR = L(\mathbf{p}(\cdot), \mathbf{n})$

Therefore, similar to the “black-box” testing case (1), the classic confidence region $D_0$ for $\mathbf{p}(\cdot)$ is defined by

$$D_0 = \{\mathbf{p}(\cdot) : L(\mathbf{p}(\cdot), \mathbf{n}) \geq \alpha\}, \tag{11}$$

where $1 - \alpha$ is the confidence level.

The confidence region $D_0$ defines the set of parameter values $\mathbf{p}(\cdot)$, which are in acceptable agreement with the observed number of failure-free component tests $\mathbf{n}$.

[1] Hardware failure and wearout mechanisms that occur in physical components need to be addressed separately.

To be conservative, we need to identify the distribution $\mathbf{p}(\cdot)$ that produces the greatest attainable bound, $q_s$, for the system *pfd* $Q$ (7) within the confidence region constraint (11), i.e.:

$$q_s = \max_{\mathbf{p}(\cdot)\in D_0} Q. \tag{12}$$

As the constraint equation for the $D_0$ confidence region is non-linear, finding the exact value $q_s$ is a non-linear optimization problem where numerical solving techniques have to be used to identify the set of values for $\mathbf{p}(\cdot)$ that maximize $Q$.

Appendix A.1 shows that there is a conservative, linear approximation to the confidence region, $D_1$, that can be used to derive an analytic upper bound $q_u > q_s$ based on the *bin-filling* algorithm reported in [20, 21]. This analysis shows that the upper bound $q_u$ can always be obtained by assigning all the failure probability to the cutset with the smallest total of component tests.

This is illustrated in Table 4 for a 2oo3 structure where the components are ordered by the number of tests (with $c_1$ having the least number of tests).

| $\mathbf{p}(\cdot)$ | $x_1p(\mathbf{x})$ | $x_2p(\mathbf{x})$ | $x_3p(\mathbf{x})$ | $\varphi(\mathbf{x})p(\mathbf{x})$ |
|---|---|---|---|---|
| $1-q_u$ | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| $q_u$ | $q_u$ | $q_u$ | 0 | $q_u$ |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 1 | $q_u$ | $q_u$ | 0 | $q_u$ |

Table 4: 2oo3 structure: worst case probability distribution $\mathbf{p}(\cdot)$

In the approximation, the $D_1$ confidence region constraint (11) for $m$ components is

$$\sum_{i=1..m} q_i n_i \leq \ln(1/\alpha). \tag{13}$$

For the example in Table 4, it follows that

$$q_u(n_1+n_2) = \ln(1/\alpha). \tag{14}$$

Appendix A.1 provides a general proof that, for any structure, the worst case confidence bound value is

$$q_u = \min\left(\frac{\ln(1/\alpha)}{N_{min}}, 1\right), \tag{15}$$

where $N_{min}$ is the smallest total of component tests in a cutset that can cause system failure.

This component-based confidence bound equation is very similar to equation (3) that is used for a full system black-box test – the only difference is that $N_{min} \leq N$.

Stated formally, $N_{min}$ is determined by the structure function $\varphi$ and the $\mathbf{n}$ tests applied to the individual components, i.e.

$$N_{min} = \min_{\mathbf{x}\in B^m:\varphi(\mathbf{x})=1} (\mathbf{x}\cdot\mathbf{n}), \tag{16}$$

where $(\mathbf{x}\cdot\mathbf{n})$ is the scalar product of the state and test plan vectors, i.e. $\sum_i x_i n_i$.

From the definition of $N_{min}$ in equation (16), it is clear that the minimum value of $(\mathbf{x}\cdot\mathbf{n})$ will always be determined by a state $\mathbf{x}$ that represents a minimal cutset. This is because any state $\mathbf{x}'$ that represents a superset of a minimal cutset $\mathbf{x}$ would only increase the number of associated component tests, i.e. $(\mathbf{x}'\cdot\mathbf{n}) \geq (\mathbf{x}\cdot\mathbf{n})$.

This is illustrated in Figure 2 for a 2oo3 vote structure. All the shaded areas represent failure states $\mathbf{x}$ where $\varphi(\mathbf{x}) = 1$. The minimal cutsets are the states with just two failed components, with the smallest sum of tests being $N_{min} = (n_1 + n_2)$. Clearly the number of tests associated with the central (non-minimal) cutset can never be less than $(n_1 + n_2)$. So the failure state with the smallest number of tests, $N_{min}$, will always be one of the minimal cutsets.

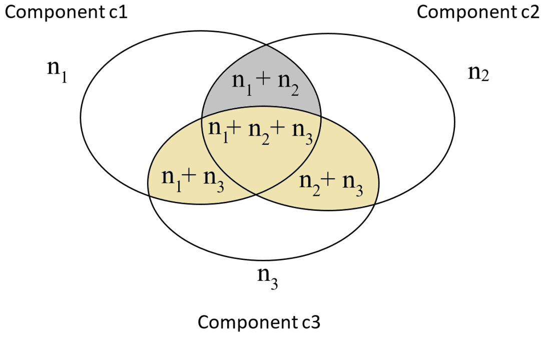

Figure 2: Example 2oo3 structure: system failure cutsets and applicable component tests

It may seem counter-intuitive that the *pfd* bound for the system is determined by a single minimal cutset while excluding other minimal cutsets completely. For example, if all component tests were equal, one might expect all failure states $\mathbf{x}$ with the same $\mathbf{x}\cdot\mathbf{n}$ value have to be included to obtain the maximum value for $q_u$. This option is allowed by the "bin-filling" algorithm described in Appendix A, but the probability $p(\mathbf{x})$ assigned to each failure state "bin" $\mathbf{x}$ is constrained by the confidence bound (13), so the nett result is the same as assigning all of the probability to a single failure state "bin".

This is illustrated in Table 5 where three components have an equal number of tests $n$, and $p(\mathbf{x}) = q_u/3$ for all three minimal cutsets. From (13):

$$\frac{2q_u}{3}(n+n+n) = q_u.2n = \ln(1/\alpha). \tag{17}$$

So $q_u$ has exactly the same bounding value as the single

| $\mathbf{p}(\cdot)$ | $x_1 p(\mathbf{x})$ | $x_2 p(\mathbf{x})$ | $x_3 p(\mathbf{x})$ | $\varphi(\mathbf{x}) p(\mathbf{x})$ |
|---|---|---|---|---|
| $1 - q_u$ | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| $q_u/3$ | $q_u/3$ | $q_u/3$ | 0 | $q_u/3$ |
| $q_u/3$ | $q_u/3$ | 0 | $q_u/3$ | $q_u/3$ |
| $q_u/3$ | 0 | $q_u/3$ | $q_u/3$ | $q_u/3$ |
| 0 | 0 | 0 | 0 | 0 |
| 1 | $2q_u/3$ | $2q_u/3$ | $2q_u/3$ | $q_u$ |

Table 5: 2oo3 structure: probability assigned to three cutsets

cutset case (14) – in both cases the confidence bound $q_u$ is determined by $2n$ component tests.

### *5.3. Accuracy of the Confidence Bound*

Appendix A.2 derives an analytic lower limit, $q_d$, to the exact confidence bound where

$$q_d = 1 - \alpha^{1/N_{min}} \leq q_s. \tag{18}$$

The difference between the lower bound, $q_d$ (18) and upper bound, $q_u$ (15) represents the maximum error in using the approximate confidence bound. This is illustrated in Figure 3 for a 95% confidence bound.

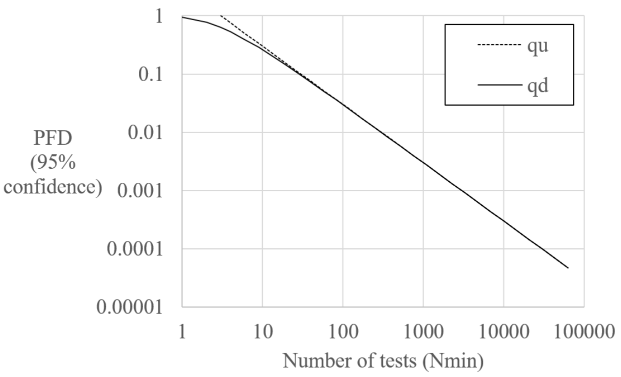

Figure 3: Comparison of upper and lower bound approximations

The error for $q_u$ can be large for a small number of tests, and the confidence bound approximation will even attain unity if $N_{min} < \ln(1/\alpha)$ which is around 3 tests for a 95% confidence bound.

However for typical statistical testing campaigns, $N$ will be more than a thousand and the confidence bound will be over-estimated by no more than 1%.

For the remainder of this paper, we will assume that the number of tests for a component always exceeds $\ln(1/\alpha)$ and hence the upper bound reduces to:

$$q_u = \frac{\ln(1/\alpha)}{N_{min}}. \tag{19}$$

## 6. Optimizing Component Tests

Given a fixed number of tests $N$, we also need to consider the optimal apportionment of tests to individual components. To minimize $q_u$, we need to identify the maximum $N_{min}$ value achievable given the constraint on the total number of tests $N$.

If all components fail identically, there is no need for apportionment as there is only one feasible cutset where:

$$N_{min} = \sum_{i=1}^{m} n_i = N \tag{20}$$

and the tests can be arbitrarily apportioned to individual components.

For symmetrical structures, the optimal apportionment is an equal number of tests for each component.

For asymmetric structures, such as the reliability block diagram (RBD) shown in Figure 4, explicit optimization is needed to ensure that the maximum value of $N_{min}$ is obtained.

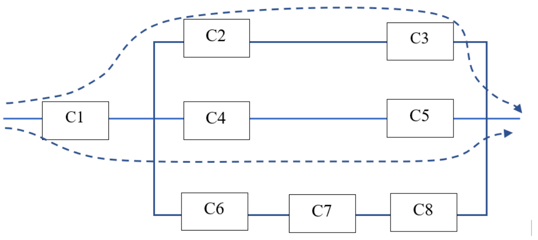

Figure 4: Shortest success paths in a reliability block diagram

Appendix B shows that it should always be possible to construct an allocation of component tests such that:

$$N_{min} \geq N/k_p. \tag{21}$$

where $k_p$ is the length of the shortest possible success path.

For example, in Figure 4, the dashed lines denote the shortest success paths where $k_p = 3$.

Appendix B also identifies two sub-optimal allocation plans that always satisfy this constraint:

- *Single shortest path*, where $N/k_p$ tests are allocated equally to all components on just one success path.
- *Balanced shortest path*, where the number of tests per component is proportional to the number of shortest success paths that include the component.

Both allocation methods are optimal for cases where each component appears only once in the RBD. The balanced path test plan also produces the optimal result for symmetric $r$ out of $m$ structures (where the same component is present in more than one RBD branch).

Figure 5 shows the result of applying the balanced path allocation procedure to the RBD in Figure 4. Components

$c_6$, $c_7$ and $c_8$ have no tests, and $c_1$ has twice as many tests as the other shortest path components (because it is present in two paths). It can be seen that the total number of tests for the components in all minimal cutsets[1] is the same ($N/3$).

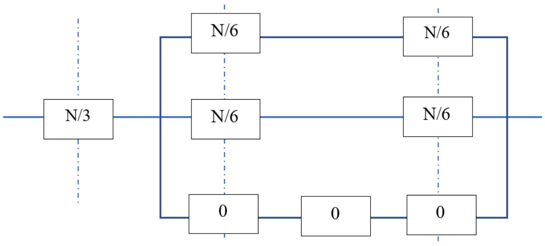

Figure 5: Allocation of tests to components

## 7. Application of the Theory to Different Structures

In this section, we apply the theory to some commonly used symmetrical structures and derive system *pfd* confidence bound formulae for different allocations of $N$ statistical tests to the system components.

Application of the theory to arbitrary asymmetrical structures is also considered.

### *7.1. Application to $r$ out of $m$ Vote Structures*

From equation (19) we have to derive $N_{min}$ to determine the system confidence bound $q_u$.

Furthermore, Section 5.2.2 shows that it is only necessary to compute the component test totals for the minimal cutsets within the structure $\varphi$ to derive $N_{min}$.

For a symmetrical $r$ out of $m$ structure, a further simplification of this derivation is possible. We know that the minimal cutset size $z$ that causes system failure is always:

$$z = m - r + 1 \tag{22}$$

For example, the number of component failures needed to defeat a 2oo3 vote is $z = 2$ (i.e. $3 - 2 + 1$).

In practical terms, this means that $N_{min}$ can be computed by ordering the component tests in a sequence $j = 1 \ldots m$ where $n_{j+1} \geq n_j$, then summing the first $z$ terms, i.e.

$$N_{min} = n_{min}(z) = \sum_{j=1\ldots z} n_j. \tag{23}$$

For the optimal case where the $N$ tests are split evenly between the $m$ components

$$N_{min} = z\frac{N}{m}, \tag{24}$$

and the system confidence bound is:

$$q_u|\varphi_{r_{oo}m} = \frac{m}{zN}\ln{(1/\alpha)}. \tag{25}$$

It follows that for a 1 oo $m$ structure where $z = m$, equation (25) reduces to:

$$q_u|\varphi_{1_{oo}m} = \frac{1}{N}\ln{(1/\alpha)}. \tag{26}$$

This is exactly the same as equation (3) – the bound achieved by statistically testing the entire system as a black-box. In addition, since

$$N_{min} = n_{min}(m) = \sum_{j=1\ldots m} n_j = N, \tag{27}$$

there is no need to split the tests evenly between components; any apportionment of $N$ tests in a 1 oo $m$ structure produces the same confidence bound.

At the other extreme, a $m$ oo $m$ structure is a series connected structure of $m$ components where any failure can cause system failure, i.e. $z = 1$. Hence for a series connected structure, the optimal equation (25) reduces to

$$q_u|\varphi_{m_{oo}m} = \frac{m}{N}\ln{(1/\alpha)}. \tag{28}$$

In addition, since the minimal cutset size is $z = 1$ for a series-connected structure, the $N_{min}$ equation for an arbitrary split of tests (23) reduces to:

$$N_{min} = n_{min}(z) = \min_{i=1\ldots m} n_i. \tag{29}$$

In the cases presented above, it is presumed that all software components are diverse, so all failure states permitted by the $\varphi_{r_{oo}m}$ structure function are feasible. With identical software-based components, the $r$ out of $m$ vote provides no defence against systematic faults as all components will fail on the same input demands. To model this software failure dependency, an additional filter function $\phi_S(\mathbf{x})$ has to be included in the structure definition where:

$$\begin{aligned}
&\phi_S(\mathbf{x}) = 1,\ \text{if } \textstyle\sum x_i = m\\
&\phi_S(\mathbf{x}) = 1,\ \text{if } \textstyle\sum x_i = 0\\
&\phi_S(\mathbf{x}) = 0,\ \text{otherwise.}
\end{aligned}$$

Combined with the architecture structure function:

$$\varphi(\mathbf{x}) = \varphi_{r_{oo}m}(\mathbf{x})\phi_S(\mathbf{x}), \tag{30}$$

we obtain:

$$\begin{aligned}
&\varphi(\mathbf{x}) = 1,\ \text{if } \textstyle\sum x_i = m\\
&\varphi(\mathbf{x}) = 0,\ \text{otherwise.}
\end{aligned}$$

It follows that the only feasible failure state is the $m$-component cutset where:

$$N_{min} = \sum_{i=1\ldots m} n_i = N. \tag{31}$$

[1] The dashed lines in Figure 5 are only example cutsets – there are further minimal cutset component combinations.

Hence:

$$q_u|\varphi_{r\text{oo}m}, \phi_S = \frac{1}{N}\ln(1/\alpha). \tag{32}$$

Since the total number of tests, $N$, determines the system confidence bound, it does not matter how the tests $n_i$ are apportioned between the components.

The results obtained from applying the theory to different symmetrical structures are summarized in Table 6.

| Structure | Software | $N_{min}$ | Optimum value | |
|---|---|---|---|---|
| | | | $n_i$ | $q_u$ |
| Series ($m$ oo $m$) | Diverse | $\min(n_i)$ | $\frac{N}{m}$ | $\frac{m}{N}\ln\frac{1}{\alpha}$ |
| Parallel ($r$ oo $m$) | Diverse | $n_{min}(z)$ | $\frac{N}{m}$ | $\frac{m}{zN}\ln\frac{1}{\alpha}$ |
| Parallel ($r$ oo $m$) | Identical | $N$ | any | $\frac{1}{N}\ln\frac{1}{\alpha}$ |
| Parallel (1 oo $m$) | Either | $N$ | any | $\frac{1}{N}\ln\frac{1}{\alpha}$ |

Table 6: Summary of demand-based confidence bound bounds

For diverse components, $n_{min}(z)$ can be derived by summing the $n_i$ values of the first $z$ components (ordered by the number of tests).

For identical components subjected to tests from the same input distribution, $N_{min} = N$ for all $r$ and $m$ values.

### *7.2. Application to Arbitrary Structures*

In Section 5.2, it was shown that we only need to consider the minimal cutsets of a structure to determine $N_{min}$. It follows that for any arbitrary structure, the value of $N_{min}$ can always be determined by the following procedure:

- identify all the minimal cutsets,
- sum the number of tests for the components in each minimal cutset,
- take the minimum of the cutset test totals.

This value of $N_{min}$ can be inserted into equation (19) to derive the system confidence bound.

## 8. Numerical Examples

The following numerical examples illustrate the derivation of the system confidence bound based on tests of individual components for cases where the total number of tests $N = 3000$ and the $(1 - \alpha)$ confidence level is 95%. The examples include symmetrical majority vote structures where an even allocation of tests is always optimal, and asymmetric structures where an uneven allocation of component tests is needed to optimize the *pfd* confidence bound that can be claimed for the system.

### *8.1. Majority Vote Examples*

Let us take as an example, a protection system where three divisions contain identical software and the division outputs are subjected to an external 2oo3 hardware vote. As the divisions are identical, they always fail simultaneously so all failures are assumed to occur in the central failure cutset (see Figure 2). So, in principle, it should not matter how the tests are distributed between divisions.

Table 7 compares the worst case bound defined by equation (32) for two alternative test plans. In the first plan the tests are equally spread between the three divisions. In the second plan, most tests are applied to a single division with only 100 tests applied to the other hardware divisions when they become available later in the project. The results are shown in Table 7 where we compare the system confidence bound derived using the analytic approximation (19), $q_u$ against the exact solution, $q_s$ derived by numerically solving equation (12).

| Component $i$ | Demands $n_i$ | Demands $n_i$ |
|---|---|---|
| 1 | **1000** | **100** |
| 2 | **1000** | **100** |
| 3 | **1000** | **2800** |
| $N_{min}$ | **3000** | **3000** |
| $q_u$ (approx.) | 0.0009986 | 0.0009986 |
| $q_s$ (exact) | 0.0009981 | 0.0009981 |
| Error (%) | +0.05% | +0.05% |

Table 7: Confidence bound comparison (2oo3 vote of identical components)

In this example, the tests for all components (highlighted in bold) are included in the $N_{min}$ value used to derive the confidence bound. It can be seen that the numerical bound and the approximated analytic bound are in close agreement and the results are not affected by the apportionment of tests between components.

Now let us apply these test plans to a 2oo3 vote system where the division software is not identical (e.g. the software is intentionally diverse or includes different ancilliary software functions). The results are shown Table 8.

| Component $i$ | Demands $n_i$ | Demands $n_i$ |
|---|---|---|
| 1 | **1000** | **100** |
| 2 | **1000** | **100** |
| 3 | 1000 | 2800 |
| $N_{min}$ | **2000** | **200** |
| $q_u$ (approx.) | 0.0014979 | 0.0149787 |
| $q_s$ (exact) | 0.0014971 | 0.0148670 |
| Error (%) | +0.05% | +0.75% |

Table 8: Confidence bound comparison (2oo3 vote of non-identical components)

In this example, the components are ordered by the number of tests performed. As the minimal cutset size is $z = 2$ for a symmetrical 2oo3 vote structure, $N_{min} = n_{min}(z)$, and this value can be determined by summing the tests for the first two components (highlighted in bold). The best result is obtained with an even split of tests, where $N_{min}$ is 2000 and the bound is 1.5 times greater than the bound derived in Table 7. For the second test plan, the sum of the two smallest component tests, $N_{min}$, is 200. In this case the confidence bound is 15 times greater than the bound derived in Table 7.

Comparing the bounds for an even split of tests on non-identical components (25) and black-box tests (3), it can be seen that, for any $r$ oo $m$ structure there is a simple ratio of bounds, $S = m/z$, that applies for any confidence level and any number of tests. Bound scale factors, $S$, are shown in Table 9 for some typical $r$ oo $m$ vote structures containing non-identical components.

| Structure | $m$ | $z$ | S $(m/z)$ |
|---|---|---|---|
| 1 oo 2 | 2 | 2 | 1.00 |
| 2 oo 3 | 3 | 2 | 1.50 |
| 2 oo 4 | 4 | 3 | 1.33 |
| 3 oo 4 | 4 | 2 | 2.00 |

Table 9: Confidence bound scale factors: component vs. black-box (even split of component tests, non-identical components)

In order to match the bound achieved with $N$ tests on a full black-box system test, majority voted components with non-identical software would need $S.N$ tests evenly split between the components. For example, the 2 oo 3 structure would need $1.5N = 4500$ component tests rather than the 3000 tests needed for a black-box.

For 1 oo $m$ structures and all $r$ oo $m$ vote structures using identical components, $S = 1$ for any split of components tests, i.e. it is as effective as black-box testing.

### *8.2. Asymmetric Structure Examples*

Section 6 showed that for an arbitrary structure function $\varphi(\mathbf{x})$, an asymmetric test plan can give the best conservative confidence bound for the system *pfd*.

For instance, let us consider the reliability block diagram shown in Figure 6.

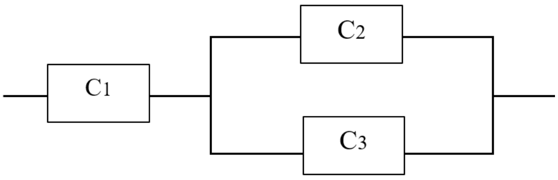

Figure 6: Example reliability block diagram

In this structure, there are two minimal cutsets $\mathbf{x}$ for the components: $\{c_1\}$ and $\{c_2, c_3\}$.

Using the procedure described in Section 7.2, Table 10 derives $N_{min}$ for these minimal cutsets using three different test plans $\mathbf{n}$ that all sum to $N = 3000$. We also calculate $q_u$ for $\alpha = 0.05$ (the 95% confidence bound).

| $\mathbf{x}$ | Tests $\mathbf{n}$ | $\mathbf{x} \cdot \mathbf{n}$ | $N_{min}$ | $q_u$ |
|---|---|---|---|---|
| 1,0,0 | 2500, 250, 250 | 2500 | | |
| 0,1,1 | 2500, 250, 250 | 500 | 500 | 0.006 |
| 1,0,0 | 1000, 1000, 1000 | 1000 | 1000 | 0.003 |
| 0,1,1 | 1000, 1000, 1000 | 2000 | | |
| 1,0,0 | 1500, 750, 750 | 1500 | 1500 | 0.002 |
| 0,1,1 | 1500, 750, 750 | 1500 | | |

Table 10: $N_{min}$ and $q_u$ derived from minimal cutsets for different test plans

The first test plan is an example where a higher order cutset determines the system *pfd* bound because it has the least combined number of component tests.

The second test plan has an even split of tests between components. This is optimal for symmetrical structures, but for the example structure, the third, asymmetrical test plan ($n_1 = 1500, n_2 = n_3 = 750$) produces a larger $N_{min}$ value and hence the smallest value for $q_u$. The third test plan uses the balanced path test plan described in Section 6 where component tests are apportioned based on the number of minimum length success paths that include the component (and $c_1$ is present in both success paths).

A further, more extreme, test allocation example can be shown for the reliability block diagram in Figure 7.

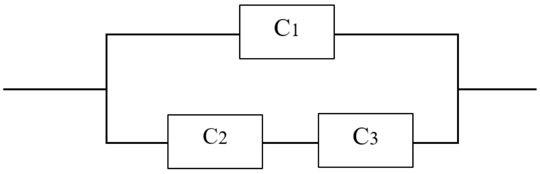

Figure 7: Alternative reliability block diagram

As the minimum length success path only includes $c_1$, the optimum test plan allocates all $N$ tests to component $c_1$, i.e. $n_1 = N$ and $n_2 = n_3 = 0$. In this allocation, the number of tests in minimal cutsets $\{c_1, c_2\}$ and $\{c_1, c_3\}$ is $N$, so $N_{min} = N$. For $N = 3000$ and $\alpha = 0.05$, the upper confidence bound is:

$$q_u = \frac{\ln(1/\alpha)}{3000} = 0.001. \qquad (33)$$

This optimized test plan produces a system *pfd* bound equivalent to testing the whole system with the same number of tests.

By contrast, with an even allocation of tests, $N_{min} = 2000$ for both cutsets, we could only claim a bound of $q_u = 0.0015$.

## 9. Discussion

### *9.1. Comparison with Conventional Failure Analysis*

The system is modelled using a structure function to identify the minimal cutsets $M$, so the system *pfd* is:

$$pfd = \sum_{\mathbf{x} \in M} p(\mathbf{x}), \tag{34}$$

where $M$ is a set of minimal cutsets, and $p(\mathbf{x})$ the failure probability of cutset $\mathbf{x}$.

This follows the standard approach used in conventional hardware analysis (PSA/PRA) [12], but our software failure model uses a different quantification approach for the $p(\mathbf{x})$ values.

In an ideal PRA, with known hardware component failure probabilities, $q_i$, that are independent:

$$p(\mathbf{x}) = \prod_{i:x_i=1} q_i. \tag{35}$$

In practice, component failures may not be independent due common cause failure (CCF) which increases the cutset failure probability. This dependency can be quantified by a range of different methods [10, 22, 23, 24] where some proportion of component failures $q_i$ are deemed to be common with the failures of the other components in the cutset (e.g. typical Beta factors [10] range between 0.01 and 0.3).

With our software failure model, a minimal cutset represents a common failure region where the cutset components *always fail concurrently*, i.e. with maximum failure dependency.

In addition, as zero failures have been observed, there are *no definitive probability values* for the individual software components, $q_i$, or individual cutsets $p(\mathbf{x})$. As a result, we are free to make the worst case assignment of software failure that maximizes the system *pfd* within the confidence region constraints.

As a result, the worst case assignment in our model occurs when all $p(\mathbf{x}) = 0$ apart from the minimal cutset with the smallest total of tests. This is because our model is seeking a worst case bound estimate rather than a realistic PRA "best estimate" (where all minimal cutsets make a quantified contribution).

### *9.2. Impact of diversity*

In hardware reliability modelling, diversity is generally considered to improve reliability by reducing the likelihood of common cause failure. For example, diversity in redundant channels is one of the factors taken into account in IEC 61508 Beta factor estimation [10].

By contrast, in our software reliability bound model, diversity can *increase* the system *pfd* bound. This is because the use of common software components eliminates some failure states in the structure function, so reducing the scope for maximization.

If there was an extra constraint in our model on the proportions of common and non-common failures for diverse components, the system *pfd* could be reduced. However it would be difficult to justify the inclusion of this extra constraint without further research.

### *9.3. Applicability of the Software Failure Model*

The statistical test theory presented in this paper is applicable to arbitrary structures. For example, it could be applied to the minimal cutsets generated by conventional reliability analysis techniques such as fault tree analysis [25]. The only difference is that, from a software based perspective, we have to combine component test results for each cutset, then use the cutset with the smallest combined number of tests to compute the worst case confidence bound for the software within the system.

The worst case software bound derived using our model can be combined with a conventional cutset calculation for hardware failure probabilities.

### *9.4. Optimization of Component Tests*

If there is scope for specifying the component tests in advance, the test plans can be optimized to minimize the *pfd* confidence bound for the overall system. Test plan optimization can produce some surprising results for asymmetrical structures – some components are not tested at all because greater benefit can be gained by assigning the tests to other more critical components.

We have identified a minimum optimization performance level for test plans that should be attainable for any structure. We also present two test plan approaches that meet this criterion. For the example structures used to illustrate these approaches, the test plans are optimal. However these approaches are not guaranteed to be optimal for all structures, and further research is needed on test plan optimization.

## 10. Summary and Conclusions

This paper considers how statistically realistic tests on components can be used to infer a confidence bound for the software within the overall system. This has practical applications for large complex systems where extensive black-box testing is infeasible before deployment.

The paper presents a general analytic method for deriving a confidence bound for an arbitrary system architecture from successful tests on individual components. This is a conservative approximation to the true confidence bound that can only be solved numerically.

In addition, two sub-optimal test allocation methods have been devised to optimize the apportionment of component tests for a given architecture.

Analytic bound equations are derived for a number of commonly used architectures and the accuracy of the analytic bound relative to the true bound is illustrated with numerical examples.

These analyses show that:

- In order to derive a conservative confidence bound for the system, there is no need to derive confidence bounds for individual components or presume independence of their failures.
- Component tests can achieve the same confidence bound as complete black-box system tests for the following architectures:
  - Any 1 out of $m$ vote structure.
  - Any $r$ out of $m$ vote structure composed of identical components.

  In both cases, any split of component tests is allowed.
- In general, the presence of non-identical components within any structure increases the system *pfd* bound relative to an equivalent structure with identical components. We showed that:
  - For evenly tested, non-identical components in a voted $r$ out of $m$ structure, the confidence bound is $m/(m-r+1)$ times the bound that can be claimed in a black-box system test.
- An even apportionment of tests across components is not necessarily the optimal choice for asymmetric structures. In some cases, the optimal apportionment of tests can be extremely asymmetric (for example, some components may not be tested at all).

While the approach has been illustrated using fairly simple architectures, it can be used in conjunction with any method that produces minimal cutsets (like fault tree analysis) to analyse the impact of software-based components within the architecture.

**Acknowledgements**

The authors wish to thank Dr. David Wright (City University) for his detailed review of this paper.

## Appendix A. Confidence Bound Approximation

### *Appendix A.1. Upper bound for $q_s$*

To derive an upper bound for $q_s$, we use the known inequality [26] that:

$$1 - x \leq \exp(-x). \tag{A.1}$$

It follows that the likelihood of a given set of $\mathbf{p}$ values is bounded by

$$L(\mathbf{p}(\cdot)) \leq \exp\left(-\sum_{i=1}^{m}\sum_{\mathbf{x}\in B^m} x_i n_i p(\mathbf{x})\right), \tag{A.2}$$

which reduces to

$$L(\mathbf{p}(\cdot)) \leq \exp\left(-\sum_{\mathbf{x}\in B^m} n(\mathbf{x}) p(\mathbf{x})\right), \tag{A.3}$$

where

$$n(\mathbf{x}) = \sum_{i=1}^{m} x_i n_i = \mathbf{x}\cdot\mathbf{n}. \tag{A.4}$$

This allows us to define an approximate confidence region:

$$D_1 = \left\{\mathbf{p}(\cdot) : \sum_{\mathbf{x}\in B^m} n(\mathbf{x})p(\mathbf{x}) \leq \ln(1/\alpha)\right\}, \tag{A.5}$$

such that $D_0 \subseteq D_1$. This approximate confidence region D1 can be used to derive a conservative confidence bound, $q_u$, for the system *pfd*

$$q_s \leq q_u = \max_{\mathbf{p}(\cdot)\in D_1} (pfd(\varphi, \mathbf{p}(\cdot))). \tag{A.6}$$

Since equation (A.5) is linear, finding the conservative confidence bound (A.6) is a linear programming problem, which can be solved with the *bin-filling* algorithm, reported in [20, 21]:

1. Order the system states $\mathbf{x} \in B^m$ in terms of decreasing $\varphi(\mathbf{x})/n(\mathbf{x})$ value.
2. While
$$\sum_{\mathbf{x}\in B^m} n(\mathbf{x})p(\mathbf{x}) \leq \ln(1/\alpha) \tag{A.7}$$
and
$$\sum_{\mathbf{x}\in B^m} p(\mathbf{x}) \leq 1, \tag{A.8}$$
assign upper limit to each probability $p(\mathbf{x})$, in turn.
3. For the remaining cutsets, set the $p(\mathbf{x})$ values to 0.

The system state probabilities $p(\mathbf{x})$ are normalized, i.e.

$$\sum_{\mathbf{x}\in B^m} p(\mathbf{x}) = 1. \tag{A.9}$$

As a result, for this particular bin-filling example, only the "bin" for at most one state can be filled up to 1. It follows from (A.7) that the confidence bound is determined by the system failure cutset "bin" that has the smallest combined number of tests where:

$$n(\mathbf{w}) = \min_{\mathbf{x}\in B^m:\varphi(\mathbf{x})=1} n(\mathbf{x}) = N_{min}, \tag{A.10}$$

and

$$p(\mathbf{w}) = \min\left(\frac{\ln(1/\alpha)}{n(\mathbf{w})}, 1\right); \tag{A.11}$$

$$p(\mathbf{x}) = 0, (\forall\mathbf{x})\ \mathbf{x} \neq \mathbf{w}, \varphi(\mathbf{x}) = 1. \tag{A.12}$$

Hence

$$q_u = \min\left(\frac{\ln(1/\alpha)}{N_{min}}, 1\right). \tag{A.13}$$

For the special case where there are several bins $\mathbf{w}$ that have an identical number of tests, it would be possible to partially fill all these bins, but as equation (A.7) is linear it makes no difference to the worst case confidence bound value. As a result, the worst case bound can still be derived by filling a single bin.

For an additional justification, one can also observe that system states, corresponding to minimal cutsets are only corner points of the feasibility polyhedron. In accordance with the theory of linear programming, the extreme of a linear objective function is always achieved at some corner point of the feasibility polyhedron. A least tested minimal cutset is a corner point with the maximum attainable value of the objective function.

### *Appendix A.2. Lower bound for $q_s$*

For some given system state (cutset)

$$\mathbf{z} : \mathbf{z} \in B^m, \varphi(\mathbf{z}) = 1,$$

let us consider the following distribution[2] $\mathbf{p}_z(\cdot)$:

$$p_z(\mathbf{z}) \geq 0; \tag{A.14}$$

$$p_z(\mathbf{0}) = 1 - p_z(\mathbf{z}); \tag{A.15}$$

$$p_z(\mathbf{x}) = 0, \text{ for all } \mathbf{x} \neq \mathbf{0} \text{ and } \mathbf{x} \neq \mathbf{z}. \tag{A.16}$$

With this distribution, there is only one probable cutset, $\mathbf{z}$, and one probable success state, $\mathbf{0}$, with all other states being improbable.

For such a distribution, equation (11) implies that $\mathbf{p}_z(\cdot) \in D_0$ iff

$$\begin{aligned} \prod_{i=1}^{m} (1 - z_i p_z(\mathbf{z}))^{n_i} &= \prod_{i=1}^{m} (1 - p_z(\mathbf{z}))^{z_i n_i} \\ &= (1 - p_z(\mathbf{z}))^{n(\mathbf{z})} \geq \alpha, \end{aligned} \tag{A.17}$$

where

$$n(\mathbf{z}) = \sum_{i=1}^{m} z_i n_i. \tag{A.18}$$

i.e.

$$p_z(\mathbf{z}) \leq 1 - \alpha^{1/n(\mathbf{z})}, \tag{A.19}$$

Thus, if we choose $p_z(\mathbf{z}) = 1 - \alpha^{1/n(\mathbf{z})}$, then

$$pfd(\varphi, \mathbf{p}_z(\cdot)) = p_z(\mathbf{z}) = 1 - \alpha^{1/n(\mathbf{z})}. \tag{A.20}$$

and

$$1 - \alpha^{1/n(\mathbf{z})} \leq q_s, \tag{A.21}$$

i.e. the *pfd* confidence bound for the whole system is worse than a *pfd* confidence bound for any of its cutsets. Hence

$$q_s \geq 1 - \alpha^{1/N_{min}}, \tag{A.22}$$

where $N_{min} = \min_{\mathbf{z} \in B^m : \varphi(\mathbf{z})=1}(n(\mathbf{z}))$.

## Appendix B. Sub-optimal test allocation plans

For complex structures, test plan optimization is needed to ensure the maximum $N_{min}$ value is obtained over all possible cutsets.

This requires a solution for the following integer programming problem:

$$N^*_{min} = \max N_{min} \tag{B.1}$$

where

$$N_{min} = \min_{\mathbf{x} \in \mathbf{B}^m :\ \varphi(\mathbf{x})=1} (\mathbf{n} \cdot \mathbf{x}) \tag{B.2}$$

under the constraint that:

$$(\mathbf{1} \cdot \mathbf{n}) = N. \tag{B.3}$$

While it is difficult to provide a formal proof that a test plan approach for deriving $\mathbf{n}$ is optimal for all structures, we have identified two sub-optimal test plans that guarantee attainable lower bounds for $N^*_{min}$.

Let us assume that there are $m$ system components and the number of components in the shortest success path of the system is $k_p$.

The set of $G$ shortest success paths can be represented by a $G \times m$ incidence matrix, $A$, where:

$a_{ij} = 1$ if component $i$ is on success path $j$,

$a_{ij} = 0$ otherwise.

The attainable bounds for $N^*_{min}$ are derived below for two sub-optimal test plans.

### *Appendix B.1. Single Path Test Plan*

Let us consider the following test plan. For some shortest success path $j'$ in A:

$$n_i = N \frac{a_{ij'}}{k_p},\ i = 1..m, \tag{B.4}$$

Thus, all the tests are allocated uniformly between the components of a single shortest path.

It is well known that each minimal cutset $\mathbf{x}$ contains at least one component $i$ from a shortest success path.

Hence,

$$\begin{aligned} N_{min} &= \\ &\min_{\mathbf{x} \in \mathbf{B}^m :\ \varphi(\mathbf{x})=1} (\mathbf{n} \cdot \mathbf{x}) \geq \min_{n_i > 0}(n_i) = \frac{N}{k_p} \end{aligned} \tag{B.5}$$

Thus,

$$N^*_{min} \geq N/k_p. \tag{B.6}$$

### *Appendix B.2. Balanced Test Plan*

A balanced test plan is based on the full set of $G$ shortest success paths represented by the $G \times m$ incidence matrix, $A$.

[2] The distribution models one of the possible dependencies between component failures (successes)

Let us define a vector $\mathbf{g} = (g_1, g_2, \ldots, g_m)'$, where $g_i$ is the number of shortest success paths that include component $i$, i.e.:

$$g_i = \sum_{j=1}^{G} a_{ij}. \tag{B.7}$$

and define $G_m$ as the total for all $m$ components, i.e.

$$G_m = \sum_{i=m}^{m} g_i = \sum_{j=1}^{G} \sum_{i=1}^{m} a_{ij} = k_p G. \tag{B.8}$$

Now let us form a *balanced* test plan, where:

$$n_i = N \frac{g_i}{G_m} = N \frac{g_i}{G \cdot k_p},\ i = 1..m. \tag{B.9}$$

One can see that the tests allocated to components, $n_i$, are based on the proportion of shortest paths that contain component $i$. In addition, if all the shortest success paths $1 \ldots G$ have disjoint sets of components, then $g_i = 1$ for all components (i.e. are only present on one path).

Each cutset $\mathbf{x}$ contains at least one component from each shortest success path. Therefore,

$$\mathbf{g} \cdot \mathbf{x} \geq G_0 \geq G, \tag{B.10}$$

where

$$G_0 = \min_{\mathbf{x} \in \mathbf{B}^m :\ \varphi(\mathbf{x})=1} (\mathbf{g} \cdot \mathbf{x}) \tag{B.11}$$

and for the balanced test plan $\mathbf{n}$

$$\mathbf{n} \cdot \mathbf{x} = \frac{N}{G \cdot k_p} \times (\mathbf{g} \cdot \mathbf{x}) \geq \frac{N \cdot G_0}{G \cdot k_p} \geq \frac{N}{k_p}. \tag{B.12}$$

Hence for any structure,

$$N^*_{min} \geq \frac{N \cdot G_0}{G \cdot k_p} \geq \frac{N}{k_p}. \tag{B.13}$$

So the balanced test plan (B.9) has the potential for an improved bound relative to the single path plan (B.4).

## References

[1] J. May, G. Hughes, A. Lunn, Reliability estimation from appropriate testing of plant protection software, Software Engineering Journal 10 (6) (1995) 206–218.

[2] D. Hunns, N. Wainwright, Software-based protection for Sizewell B: the regulator's perspective, in: Electrical and Control Aspects of the Sizewell B PWR, 1992., International Conference on, IET, 1992, pp. 198–203.

[3] ONR, EDF and AREVA UK EPR Generic Design Assessment: GDA Issue - Protection System Independent Confidence Building Measures, GI-UKEPR-CI-02 Revision 2 (2011).

[4] ONR, Safety Assessment Principles for Nuclear Facilities, Office for Nuclear Regulation (2014).

[5] IEC, Nuclear power plants - Instrumentation and control systems important to safety - Classification of instrumentation and control functions, IEC 61226 Edition (2009).

[6] IEC, Nuclear power plants - Instrumentation and control systems important to safety - General requirement for systems, IEC 61513 Edition (2011).

[7] IEC, Nuclear power plants - Instrumentation and control systems important to safety - Software aspects for computer-based systems performing category A functions, IEC 60880 Edition (2006).

[8] W. Ehrenberger, Statistical testing of real time software, in: Verification and Validation of Real-Time Software, Springer, 1985, pp. 147–178.

[9] D. L. Parnas, G. Asmis, J. Madey, Assessment of safety-critical software in nuclear power plants., Nuclear Safety 32 (2) (1991) 189–198.

[10] IEC, Functional safety of electrical/electronical/programmable electronic safety-related systems, ed. 2, IEC 61508:2010 (2010).

[11] ONR, Technical Assessment Guide: Computer Based Safety Systems Safety Assessment Principles for Nuclear Facilities, Office for Nuclear Regulation, NS-TAST-GD-046 Revision 5 Edition (2019).

[12] E. J. Henley, H. Kumamoto, Probabilistic risk assessment and management for engineers and scientists, 2nd Edition, Wiley-IEEE Press, 1996.

[13] S. Morris, Use and Application of MIL-HDBK-217, Journal of the IES 33 (6) (1990) 40–46.

[14] D. Cox, D. Hinkley, Theoretical Statistics, 1st Edition, Chapman & Hall/CRC, 1979.

[15] P. G. Bishop, The variation of software survival time for different operational input profiles, in: FTCS-23 The Twenty-Third International Symposium on Fault-Tolerant Computing, IEEE, 1993, pp. 98–107.

[16] J. C. Knight, N. G. Leveson, An experimental evaluation of the assumption of independence in multiversion programming, IEEE Transactions on Software Engineering 12 (1) (1986) 96–109.

[17] D. E. Eckhardt, L. D. Lee, A theoretical basis for the analysis of redundant software subject to coincident errors, IEEE Transactions on Software Engineering 11 (12) (1985) 1511–1517.

[18] B. Littlewood, D. R. Miller, Conceptual modeling of coincident failures in multiversion software, IEEE Transactions on Software Engineering 15 (12) (1989) 1596–1614.

[19] R. E. Barlow, A. S. Wu, Coherent systems with multi-state components, Mathematics of Operations Research 3 (4) (1978) 275–281.

[20] P. Bishop, R. Bloomfield, B. Littlewood, P. Popov, A. Povyakalo, L. Strigini, A conservative bound for the probability of failure of a 1-out-of-2 protection system with one hardware-only and one software-based protection train, Reliability Engineering & System Safety 130 (2014) 61–68.

[21] P. Bishop, A. Povyakalo, Deriving a frequentist conservative confidence bound for probability of failure per demand for systems with different operational and test profiles, Reliability Engineering & System Safety 158 (2017) 246–253.

[22] P. Hokstad, A generalisation of the beta factor model, in: Probabilistic Safety Assessment and Management, Springer, 2004, pp. 1363–1368.

[23] G. Apostolakis, P. Moieni, The foundations of models of dependence in probabilistic safety assessment, Reliability Engineering 18 (3) (1987) 177–195.

[24] A. Mosleh, N. Siu, A multi-parameter common cause failure model, in: Transactions of the 9th International Conference on Structural Mechanics in Reactor Technology. Vol. M, 1987.

[25] W. E. Vesely, F. F. Goldberg, N. H. Roberts, D. F. Haasl, Fault tree handbook, Tech. rep., Nuclear Regulatory Commission Washington DC (1981).

[26] H. Turnbull, An elementary derivation of the exponential limit and of Euler's Constant, Mathematical Notes 29 (1935) xxi–xxiv. doi:10.1017/S1757748900002401.