Estimating PLC logic program reliability

Peter G. Bishop

Centre for Software Reliability, City University
London, UK

Abstract.

In earlier research we developed a theory for predicting the reliability of conventional sequential programs based on an estimate of residual faults. This paper describes how the theory was applied to a realistic industrial example containing a known number of faults. The industrial example was implemented in a PLC application language where the program is represented by a network of logic gates (e.g. AND and OR gates). To make a residual fault estimate, our fault estimation method had to be adapted to apply to logic networks. The previous estimation method relied on a measurement of code coverage, and this had to be replaced by a measurement of logic network coverage. Several different measures of logic coverage were evaluated, including coverage of input values, output values, and input-output pair values. Using the residual fault estimate and information about the testing applied, a reliability bound was calculated and we assessed the sensitivity of the bound to changes in the operational profile.

1 Introduction

In earlier research we have developed a number of new approaches for estimating the reliability of software-based systems, namely:

1. A worst case reliability bound theory that can be applied to both continuous and demand-based systems [Bishop 1996, Bishop 2002a]. In its simplest form, this method only requires an estimate of the number of residual faults (*N*) and the number of tests (*T*), The theory predicts that after *T* test demands under a given test profile, the expected value of probability of failure on demand, E(PFD), will be bounded by:
   E(PFD) < N/*e*.T (where *e* is the exponential constant, 2.718…)
2. A theory for "re-scaling" the worst case bound for new operational profiles [Bishop 2002a]. One interesting result of this theory is that a "fair" test profile (where all paths are tested equally) is not very sensitive to changes in operational profile.
3. A model that relates test coverage to the number of residual faults *N* [Bishop 2002b].

4. An extension of the reliability theory to "fractional faults" [Bishop 2002a] where N<1. In this case it can be shown that the probability of surviving for T tests without failure is always greater than $(1 - N)$.

In this paper, we apply these methods to a realistic safety-related industrial PLC application. The research study objectives were to:

- Estimate the likelihood of residual faults in the PLC application logic
- Estimate the probability of failure per demand given the level of customer testing.

We first describe the industrial application used in this study, and then describe the approach used to estimate the number of residual faults and the probability of failure on demand for the logic example. This includes an analysis of the sensitivity of the estimate to changes in the operational profile.

2 The industrial example

The particular example used in this paper was taken from an earlier industrial research study where a control and protection system was re-implemented in software using a fail-safe PLC. The original logic controlled a mechanic device by means of motors and position sensors mounted on the device.

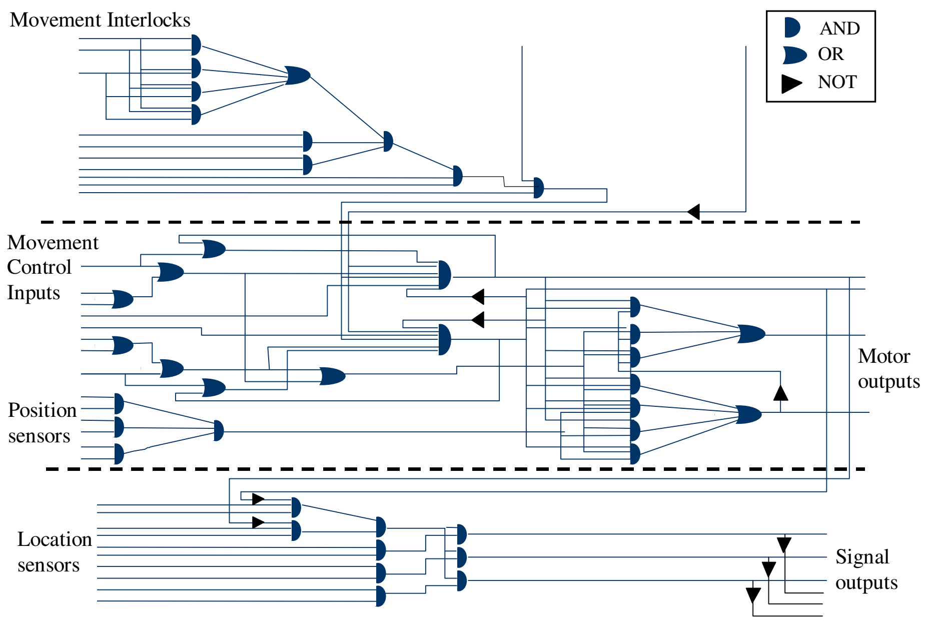

Fig. 1. Industrial control and protection logic network

As the original logic used proprietary logic circuits, the logic was translated into a logic specification. Both the customer and the PLC supplier in this study performed an independent translation. The customer translation was the basis of the oracle. The result of the customer translation is shown in the Figure 1. The logic uses AND, OR and NOT function blocks with some feedback to "latch" some of the outputs.

The supplier implementation was subjected to a range of customer acceptance tests comprising both realistic and random input sequences and compared to an independent implementation of the logic. The logic was also subjected to design review and manual testing using a switch-box.

The combination of realistic and random customer tests found 6 faults (1 interlock, 4 movement, 1 indicator signal). No faults were found in the final version, even though this was subjected to additional tests—around 10^6 random tests and around 40 000 random variations on normal operational sequences.

We used the logic specification, the set of faults found and the test specifications to evaluate our fault estimation and reliability prediction methods.

3 Residual fault estimation

To estimate the probability of failure per demand, we first require an estimate for the number of residual faults (N). A theory was developed in [Bishop 2002b] that relates the code coverage achieved with the number of residual faults. This is based on the concept of executing a "coverage element" which is some part of the code structure, e.g. a program statement, a program block between decision points, a program branch, etc. The theory assumed that:

1. A fault is equally likely to affect any one of the logic coverage elements.
2. There is a fixed probability of failure f when a coverage element is activated.
3. Faults are limited to a single coverage element.

If these assumptions apply to a measure of logic coverage, the same theory can be used. However in logic networks, assumption (3) that a fault is limited to a single coverage element may not hold. Looking at practical logic circuits, a number of different combinations of input value, K, could reveal the same fault. Applying the theory in [Bishop 2002b], it can be shown that for the case where the proportion of uncovered U decreases exponentially, the expected number of residual faults N will be:

$$N = N_0 \cdot U^{f}$$

Where N_0 is the number of faults prior to testing. It can be shown that if assumption 3 is violated and a fault spans K coverage elements, then:

$$N = N_0 \cdot U^{f \cdot K}.$$

Or if we define F as $f \cdot K$, this simplifies to:

$$N = N_0 \cdot U^{F}$$

While f cannot be greater than unity, K can exceed unity so it is possible to have $F > 1$, and in this case, the theory predicts that the proportion of faults detected can be greater than the proportion of coverage achieved.

To apply this theory to the logic example, we need to identify a suitable coverage measure for the logic, and then derive the F parameter in order to predict the number of residual faults. An ideal coverage measure, would:

- Maximise F (as this increases the rate of fault detection).
- Have a sufficiently small number of distinct coverage values to achieve 100% coverage in a reasonable time.

3.1 Logic coverage measures

The logic coverage measures that we investigated were:

1. *Input value coverage*, where all possible combinations of input values are covered.
2. *Output value coverage*, where all possible combinations of output values are covered.
3. *Input-output pair coverage*, where input values are selected such that a given input can "toggle" the state of a given output.

Complete input coverage is impractical for this logic network as it requires 2^{36} tests, but one other possibility is to use *input subset coverage*. In Figure 1, the dashed lines separate three distinct zones in the logic that only have a small number of interconnections. Given the number of inputs and interconnections between logic zones it should be possible achieve complete input coverage of the logic subsets with no more than 2^{18} tests per logic subset.

With ten binary outputs, only 2^{10} output combinations are possible for the example logic, and in practice the constraints imposed by the logic network exclude the majority of output combinations. An analysis of the intended function of the logic suggested that only 12 of the output combinations should occur. This was likely to be a relatively coarse measure for estimating the number of residual faults.

An input-output pair is defined as: *a combination of input values* $<V_1 .. V_{max}>$ *where a change of input value* V_I *will result in a change to output value* V_J. There is a strong relationship between this measure (termed I-O pair for short) and the Modified Condition Decision Coverage (MCDC) test method used in conventional programs [RTCA 93]. In MCDC testing, the values of the terms in an IF condition are selected so that a change to an individual term (e.g. from TRUE to FALSE) changes the Boolean value of the whole IF condition. The main difference when testing a logic network is that an input change can affect multiple outputs rather than the single Boolean value. To assess the coverage with respect to multiple outputs, logic coverage is measured in terms of input-output pair combinations. Note that several outputs might be toggled simultaneously when an input changes, so more than one pair might be covered by a single test. A maximum 4 input-output combinations are possible for a given *I,J* pair, so the maximum number of combinations with 36 inputs and 10 outputs is $4 \times 36 \times 10 = 1440$. In practice the number of actual combinations could much less due to constraints between inputs and outputs imposed by the logic network. The number test could be even lower as several outputs might be toggled in a single test.

As I-O pair coverage focuses on "sensitive paths", this logic coverage measure should increase the value of *F*, while the limited number tests needed to achieve coverage suggests that it should be practical to apply to realistic networks.

3.2 Coverage measurement environment

To measure the coverage achieved under testing, we implemented a coverage measurement testbed in Perl. The environment comprised several different random data test generators and predefined test sequences that could be fed into a model of the PLC logic. The logic model was instrumented to determine the coverage achieved for each measure.

The types of random test data generated were:

- uniform input random—where the probability that an input is set TRUE (pI) is 0.5 for all inputs I)
- single bit random (only one bit altered randomly on each test) but still pI = 0.5 for inputs I
- uniform output random (where the input probabilities are set so that pJ = 0.5. for all *outputs J*)

The rationale for uniform *output* probability testing is that it will maximise the output coverage growth measure, and is also likely to increase I-O pair coverage growth as there is a 50% that each output can be "toggled" by a change in an input.

In order to achieve uniform output coverage, it was necessary to devise a procedure for back-propagating assigned output probabilities to the inputs. The rules for back propagation through logic are quite simple:

AND: $p_{in} = p_{out}^{1/n}$

OR: $p_{in} = 1 - (1 - p_{out})^{1/n}$

NOT: $p_{in} = (1 - p_{out})$

where:

n is the number of inputs to the logic gate,

p_x is the probability of TRUE value on link x,

This is illustrated below for a simple single-output network, where a p value of 0.5. is back propagated to the inputs.

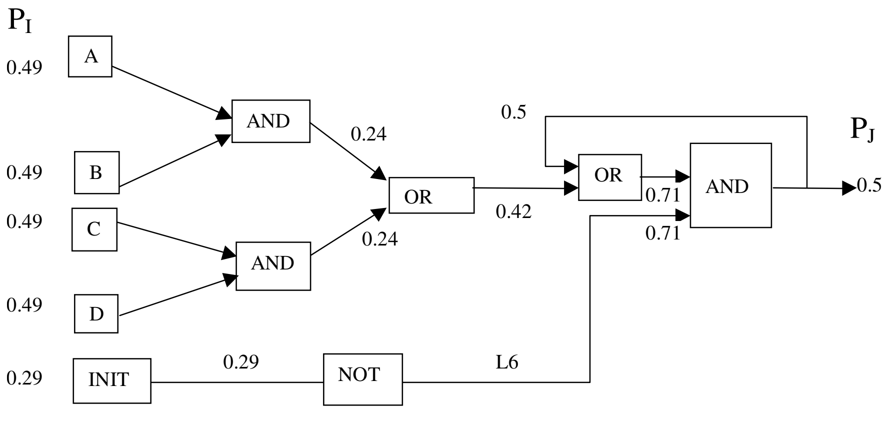

Fig. 2. Illustration of back propagation of probabilities

In practice, back propagation is constrained by network junctions and feedback loops. For example in Figure 2 above, the feedback loop forces one input to the OR gate to be 0.5. When one input probability is constrained p^*_{in}, it can be shown that he probability for the remaining inputs p_{in} is:

$p_{in} = 1 - ((1 - p_{out})/(1 - p^*_{in}))^{1/n-1}$

hence for p^*_{in}=0.5, p_{out}=0.71 and n=2, we obtain 0.42 for the other input. When back-propagation was applied to the actual network of 36 inputs, 10 outputs, the interconnection constraints meant that the "ideal value" of p_J = 0.5 could not be achieved (as negative values for p_{in} are derived during back propagation). Compromise values for the output probabilities (typically of the order of 0.3) were chosen instead to obtain valid input probabilities. The distribution of input probabilities to achieve near uniform output probabilities is shown in Figure 3. It is clear that the input probabilities can be quite extreme (close to zero or 1).

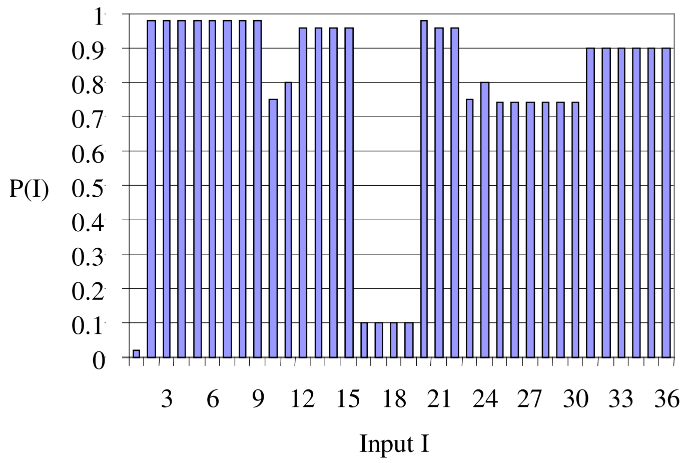

Fig. 3. Input probability distribution for near-uniform output probability

We can view this input distribution as the probabilistic equivalent of an MCDC test pattern as it maximises the chance of specifying inputs patterns where an output will change if a single input bit is changed.

3.3 Assessment of coverage growth

Using the PLC logic simulator, the coverage achieved under MCDC random and uniform random tests was measured. We also replicated some of the customer test where data was available. The coverage growth achieved under MCDC random testing is shown in Figure 4.

As can be seen from Figure 4, under MCDC random testing, full output coverage is achieved in around 100 tests, and full I-O pair coverage is reached in around 3000 tests (equivalent to 236 different I-O values). By contrast, input coverage of the logic sub-nets grows more slowly (as expected, as there are more input coverage value combinations).

The logic was retested with a uniform random input profile, the most striking changes in the growth curves were:

- Output and I-O pair coverage was much slower—the output coverage was only achieved after 106 tests, and I-O pair coverage only reached 84% after 106 tests.
- Input coverage was better—100% coverage of the interlock logic is achieved after 10^5 tests

This is consistent with expectations as uniform random input testing should maximise growth of the input coverage measure.

Coverage was also measured for the original customer "mixed" test data file of 10 000 tests comprising:

- operational sequence (159 tests)
- static interlock test data (28 test)
- dynamic interlock test data (243 tests)
- uniform random test input data (9514 tests)

The operational sequence tests proved to be the most effective means for achieving IO pair and output-coverage growth.

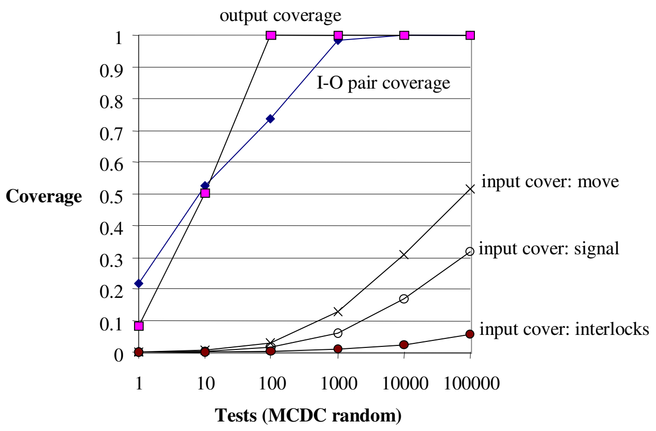

Fig. 4. Growth in coverage measure (MCDC random test profile)

3.4 Relating coverage measures to faults detected

We designed the PLC logic simulator so that the individual logic faults could be switched on and compared to the behaviour of the correct logic network when a test is applied. The cumulative fault detection counts under different test profiles are summarised in Table 1.

Table 1. Faults detected versus number of tests (different test methods)

Number of tests	Faults detected		
	MCDC test	Uniform Random	Customer tests
10	4	0	-
100	5	0	-
486	6	0	4 (ops)
1000	6	0	-
3000	6	1	-
9514	6	4	5 (random)
10000	6	4	6 (mixed)
100000	6	6	-
1000000	6	6	-

The MCDC random testing seems to be highly effective with 4 faults detected in the first 10 tests, and all 6 faults detected after 486 tests.

We then investigated whether the fault detection performance was correlated to the various types of coverage measure. Ranking input coverage against detected faults produced little obvious correlation. For example, input coverage growth was better with uniform random than MCDC, but a fault in the interlock logic was revealed 81 times in 10 000 MCDC tests and zero times in 10 000 uniform random tests.

Output coverage exhibited a better ordering between coverage measure and detected faults with different test profiles, but the coverage hit 100% before all faults were revealed. We concluded that output coverage was too coarse for predicting residual faults with any accuracy.

The best ordering was found between faults detected and I-O pair coverage. This is shown in Figure 5 above, together with the fault detection curve predicted by the model described in Section 3.

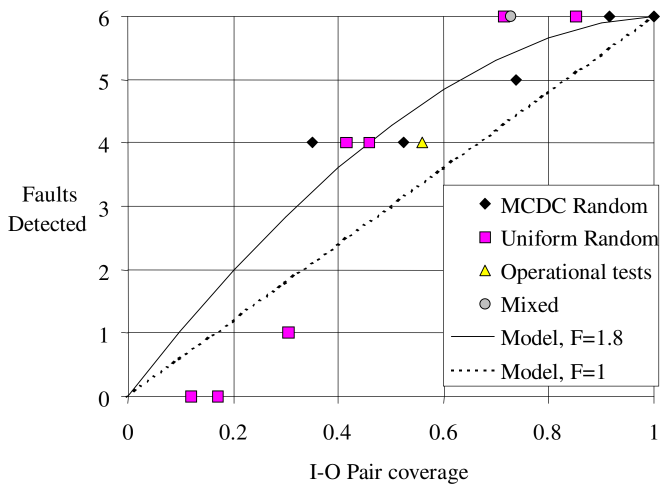

Fig. 5. I-O pair coverage vs. faults detected: comparison with model

While a simple linear model (*F*=1) is a reasonable approximation to the data for this logic network, a better fit at high coverage is achieved with a model parameter of *F*=1.8. This is in line with expectations, as we know that *F* can be greater than unity if a fault spans several coverage items. It also means that complete fault detection is likely to be achieved at lower coverage values. For example the uniform random tests found all 6 faults with only 70% I-O pair coverage.

3.5 Residual fault estimation using the model

Having parameterised the model it is possible to convert a coverage measure into a residual fault estimate. Using the logic simulator we measured the coverage achieved using the customer tests that were applied to the Intermediate version of PLC logic implementation. The customer tests are summarised in Table 2.

Table 2. Customer tests applied to Intermediate version of PLC logic

Test Type	Number of tests
Random Single Input Tests:	60 000
Total Random Tests:	160 000
Plant Operational Sequence Tests:	159

The coverage achieved by applying around 220 000 customer tests was 195 I-O pairs, i.e. the fraction of I-O pair coverage is 195/236, hence C = 0.826, so the fraction that is not covered is U = 0.174. According to our coverage model, the fraction of undetected faults is:

$$N / N_0 = U^{F}$$
$$= 0.174^{1.8}$$

Therefore:

$$N / N_0 = 0.042$$

Assuming that the 6 faults found represent N_0, the estimate for residual faults is:

$$N = 0.26$$

This is an example of the "fractional fault" case that was examined in [Bishop 2002a]. The result could also be interpreted as saying that there is at least a 74% chance of zero faults in the logic. This result should be reasonably conservative as some customer tests were not available for inclusion in the coverage measurement. A purely linear model would have predicted N=1.

By comparison, 3000 MCDC random tests achieved full coverage of 236 I-O pairs, which would result in an estimate that N is effectively zero.

4 Application of the reliability theory

4.1 Scaling reliability bounds

Our worst case bound research [Bishop 2002a] suggests that it is possible to take a reliability bound for a given test profile and scale it up for a different operational profile. The basic assumption in this theory is that, for a fault is associated with coverage element i (e.g. a particular output value), the failure rate is proportional to the element execution rate, $X(i)$. If the logic is tested under one profile and operated under another profile, it follows that the failure rate is scaled by:

$$S(\mathrm{i}) = X_{op}(i)/X_{test}(i)$$

Of course we do not know which coverage element i a fault is located in, so the mean scale factor S is the average of $S(i)$ over all coverage elements. The theory in [Bishop 2002a] shows that the reliability bound is also scaled by S when the profile changes, i.e.:

$$\mathrm{E(PFD)} < S{\cdot}N / e{\cdot}T$$

In addition the research showed that a "fair" testing profile might be derived where the reliability bound is relatively insensitive to changes in the operational profile.

If the MCDC random test profile is close to a "fair test", all coverage elements should be evenly tested. This is quite difficult to assess for I-O pair coverage where there could be constraints between I-O pair values, but analysis of the coarser output value coverage yields the following results.

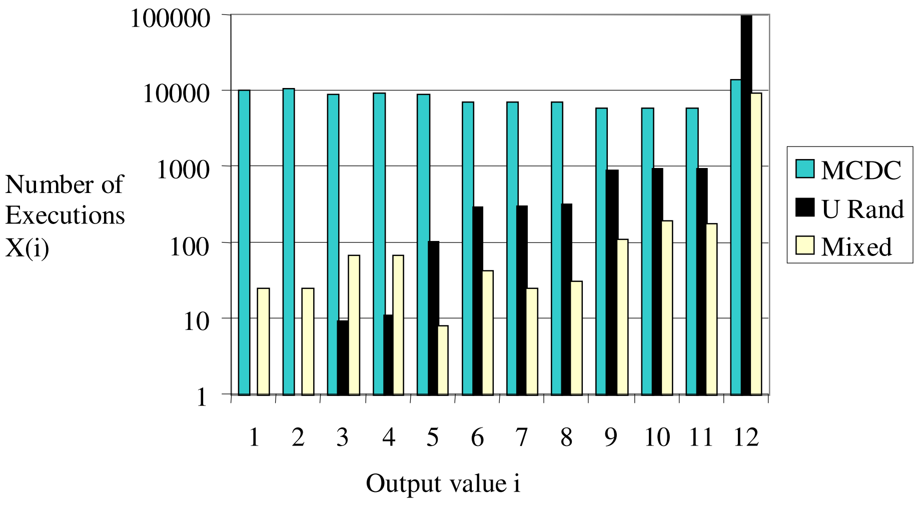

Fig. 6. Execution of output coverage elements

It is clear the MCDC random testing produces an output coverage that is close to an even distribution, while the uniform random input tests produce the most asymmetrical distribution. Assuming that a fault only affects one coverage element (a reasonable assumption for the relatively coarse output coverage), the re-scaling theory developed in [Bishop 2002a] can be applied.

Note the calculation of *S* assumes that the actual faults present in the logic are equally likely to affect any coverage element. A more pessimistic scale factor would be the mean of the *N* largest value of *S(i)* (as there are *N* faults). Similarly a more optimistic scale factor would be the mean of the *N* smallest *S*(i) values. Table 3 shows the scale factors derived from the execution rate distributions in Figure 6.

Table 3. Predicted scale-up factors for different test and operational profiles

Test Profile	Operational Profile	Scale factor		
		Min N	Mean	Max N
Uniform random	MCDC random	10.8	1893	3775
MCDC random	Uniform random	0.03	0.64	1.3
Mixed	MCDC random	6.4	25	44
MCDC random	Mixed	0.009	0.62	1.2

It can be seen that the theory predicts that the use of ‘unbalanced” profiles for testing (e.g. uniform random or mixed) will lead to very large increases in the calculated PFD bound when a dissimilar profile is used in operation. By contrast the PFD bound derived using a balanced test profile might even decrease when a dissimilar profile is used in operation.

If we take the case where uniform random data is used in testing, the theory in [Bishop 2002a] predicts that the bound on the PFD under this profile after T tests will be:

$E(PFD) \le N / e.T$

If the operational profile is MCDC random was used in operation the theory predicts that the operational PFD will lie within a rescaled bound of:

$E(PFD) \le 3775.N / e.T$

where 3775 in the pessimistic scale factor in Table 3. If the situation is reversed and MCDC is used in testing and uniform random is used in operation, the theory predicts that the operational PFD will lie within a rescaled bound of:

$E(PFD) \le 1.3.N / e.T$

where again we take the more pessimistic scale factor in Table 2.

4.2 Evaluation of the reliability bound predictions

We evaluated the bound predictions using the known faults present in the logic. The failure probability per test, *p(n)* of each fault under different profiles were measured using the logic simulator with different faults enabled. The failure rates under different profiles are shown in Table 4.

Table 4. Failure probability per test, *p(n)* under different test profiles

Fault *n*	*p(n)* MCDC	*p(n)* Uniform
1	0.008	0.00020
2	0.40	0.00014
3	0.40	0.00015
4	0.39	0.00004
5	0.39	0.00021
6	0.30	0.00002

It can be seen that MCDC random testing achieves the highest failure probabilities for each fault.

Using the measured failure probabilities for the logic defects in Table 4, reliability growth was modelled under one profile (to represent testing), then the reliability was re-computed under a different operational profile. The failure probability after *T* tests followed by a switch to new operational profile, is determined by the equation:

$E(PFD') = \sum p'(n) \cdot (1-p(n))^T$

where $p'(n)$ is the failure probability of fault *n* under the new operational profile, and $p(n)$ is the failure probability of fault *n* under the original test profile.

Figure 7 shows the reliability growth curves for these faults under the original test profile (uniform random) and the operational profile (MCDC random), together with the original and scaled bound predictions. Note that the bound predictions are not based on a knowledge of the actual logic defect failure probabilities, but only on an estimate of the number of faults (assumed to be N=6) and the execution rates of the output coverage elements under different test profiles. It can be seen that the PFD of the actual faults are below the predicted worst case bound, and the PFD changes by three orders of magnitude when the profile is changed. The PFD would have exceeded a bound based on the mean scale factor (S=1893), so the faults may disproportionately affect coverage elements with the largest S(i) values.

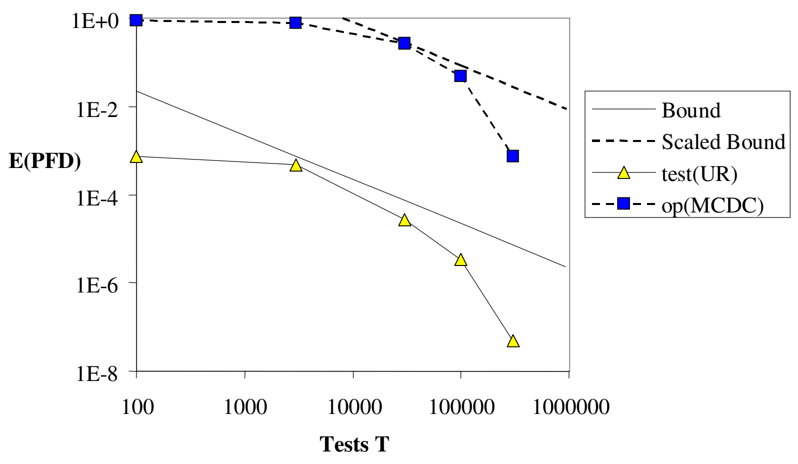

Fig. 7. Re-scaled worst case bound for a new operational profile

It is clear that the reliability prediction is highly sensitive to the test profile used. We repeated the evaluation for the converse case (MCDC random data during testing, uniform random for operation). The results are shown in Figure 8.

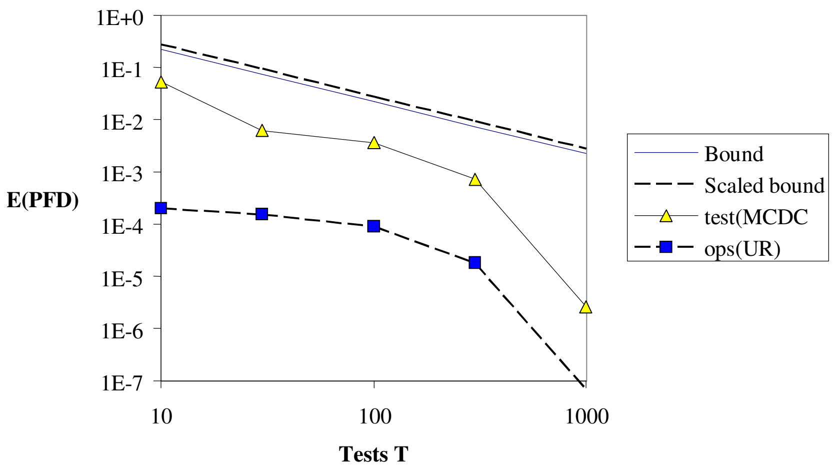

Fig. 8. Rescaled bound: after 'fair" MCDC testing

It can be seen that the PFDs lie below their respective bounds. In this case, we predicted that the scale factor for the bound would be no more than 1.3, but in fact the actual PFD *decreased* by two orders of magnitude (i.e. the scaling is less than unity). This is close to the more optimistic predicted scale factor of 0.03 (see Table 3).

4.3 Comparison of reliability predictions

From an analysis of $2.2 \cdot 10^5$ customer tests we estimated that the number of residual faults was N=0.26. What does this imply for the future reliability of the logic software? We compared the predictions of the black box Bayesian method of [Littlewood 1993] and the "worst case" reliability model presented in [Bishop 2002a].

The black box Bayesian method is based on the last failure free interval during testing T_0, so this interval will be less than $2.2 \cdot 10^5$ as 6 faults were found during testing, but we will assume $T_0 = 2.2 \cdot 10^5$ to give a best case prediction.

The worst case reliability function [Bishop 2002a] has similar in shape to the Bayesian reliability function where the probability of operating without failure $R(t|T)$ decreases almost inversely with the number of tests t. We have two ways of using the worst case theory:—using the initial value of N=6 and T=$2.2 \cdot 10^5$, or the final fractional value of N=0.26, but claim no credit for prior testing (T=0). As mentioned earlier, when the number of residual faults N is fractional, the probability of survival is asymptotic to $1-N$; for the case where N=0.26, the asymptote is 74%. The reliability predictions are shown Figure 9.

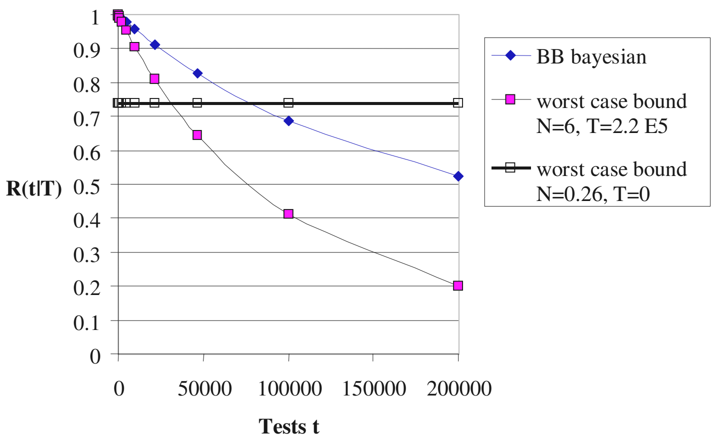

Fig. 9. Comparison of reliability predictions

The N=6 worst case reliability model predicts a 41% chance of surviving a further t=10^5 test demands, while the Bayesian method predicts a survival probability of at most 69%. However, using the N=0.26 case, the survival probability would be 74% and would remain so for any number of further demands. As the worst case functions are both bounds, it is legitimate to take the maximum of the two bound predictions. In addition, the N=0.26 prediction is unaffected by changes in operational profile. For the "mixed" test profile, scale factors of 25 in the failure rates are possible in operation (see Table 3), which would reduce the predicted survival times by a factor of 25—but this rescaling makes no difference to an asymptote.

The apparently optimistic prediction of the worst case reliability theory seems to be supported by the fact that no faults were detected by the customer when the PLC logic was retested with more than 10^6 tests after fixing the detected faults.

5 Summary and conclusions

Previously, the reliability theory had only been applied to conventional program code. However it proved to be fairly easy to adapt the theory to PLC logic networks. Minor extensions to the coverage growth theory were needed, and we also needed to identify suitable coverage measures for logic rather than conventional code. Once this had been done, it was possible to use:

1. coverage growth theory for fault estimation
2. re-scaling theory to adjust a PFD bound for a new operational profile
3. the concept of a balanced test profile to derive a PFD that is insensitive to changes in operational profile
4. the concept of a "fractional residual fault" to give less pessimistic reliability estimates than other reliability models.

It should be noted that the reliability estimation described in this paper only applies to faults in the application logic. Other faults in the PLC (e.g. in the firmware) have to be addressed separately.

Output and I-O pair coverage measures were found to correlate with detected faults and both measures might be applicable for estimating faults in other logic networks.

There is a non linear relationship between coverage and faults. A coverage growth model can be fitted to the observed data to estimate residual faults, but it is probably conservative to assume a linear relationship between coverage and faults found, and then devise a test strategy that maximises the coverage (like MCDC random testing).

We found that the "statistical MCDC" test method was the most effective for achieving high I-O pair and output coverage, while uniform random input testing was the least effective. The "statistical MCDC" method also appears to be a "fair" test profile as there is fairly balanced coverage of output values.

The scaling theory in [Bishop 2002a] predicts that a PFD bound estimate derived using a test profile that gives balanced coverage will be insensitive to changes in operational profile. This prediction was supported by simulated reliability growth calculations applied to the known faults using balanced and unbalanced tests.

We conclude that:

1. Coverage analysis theory can be used to estimate the number of residual faults in logic networks.
2. Worst case bound and rescaling theory can be used to support software reliability claims.
3. MCDC random tests should be used more widely for testing logic systems both for detecting faults and for long term reliability testing.

6 Acknowledgements

This work was funded by the UK (Nuclear) Industrial Management Committee (IMC) Nuclear Safety Research Programme under British Energy Generation UK contract 40029264 with contributions from British Nuclear Fuels plc, British Energy Ltd and British Energy Group UK Ltd. The paper reporting the research was produced under the EPSRC research interdisciplinary programme on dependability (DIRC).

References

[Bishop 1996] P.G. Bishop and R.E. Bloomfield, "A Conservative Theory for Long-Term Reliability Growth Prediction", IEEE Trans. Reliability, vol. 45, no. 4, pp. 550-560, Dec. 1996.
[Bishop 2002a] P.G. Bishop, "Rescaling Reliability Bounds for a New Operational Profile", International Symposium on Software Testing and Analysis (ISSTA 2002), (Phyllis G. Frankl, Ed.), vol. 27 (4), pp. 180-190, ACM Software Engineering Notes, Rome, Italy, 22-24 July, 2002
[Bishop 2002b] P.G. Bishop, 'Estimating Residual Faults from Code Coverage", Safecomp 2002, pp. 163-174,Catania, Italy, 10-13 Sep. 2002.
[Littlewood 1993] B. Littlewood and L. Strigini, "Assessment of ultra-high dependability for software-based systems", CACM, vol. 36, no. 11, pp.69-80, 1993.
[RTCA 1993] RTCA DO-178B , 'Software considerations in airborne systems and equipment certification", 1993.