Distinct Failure Diversity in Multiversion Software

Derek Partridge & Wojtek Krzanowski
University of Exeter, U.K.

August 8, 1997

Abstract

In earlier studies of multiversion programming, both empirical and analytical, emphasis switched from notions of independence to one of minimization of coincident failure. We show that neither independence of failure, nor lack of coincident failure are the single important properties. Indeed, an N-version system may deliver an optimal performance (under some voting strategy) even when the incidence of coincident failure is arbitrarily high. The key notion that this study contributes is one of distinct different failure, and hence distinct-failure diversity. The important property is not whether versions fail on the same input so much as whether they fail in the same way. If the failures of an N-version system (on some input) are dispersed over a set of distinct alternative outcomes, then this (hitherto unacknowledged) aspect of diversity may be exploited to substantially enhance system reliability. We propose measures for the traditional *coincident-failure diversity* (CFD), for the new *distinct failure-diversity* (DFD), and for an integrated *overall diversity* (OD). We show how the DFD property of an N-version system may permit substantial performance enhancement despite maximum coincident failure. We demonstrate the extra practical benefits that accrue from an exploitation of this new treatment of multiversion software systems by application to previously published examples. Finally, we suggest how this new aspect of diversity (as well as several others) can be exploited to cast the potential for multiversion software engineering in a much more positive light than was produced by the previous studies which demonstrated the necessary absence of independent failure behaviour.

Index Terms — Coincident failure, software diversity, distinct-failure diversity, multiversion programming, software faults, software reliability, N-version programming

1 Introduction

Multiversion, or N-version, programming has long been considered as a potential route to enhanced software reliability. It is a fault-tolerance strategy that involves separate preparation of a number of alternative versions of the

software and the application of a *decision strategy*, such as voting, to determine the overall system outcome from the collection of individual version results. However, the basis on which this redundancy is expected to provide enhanced reliability has been consistently oversimplified.

It is not simply independence of failure (as [5] assert), nor diversity with low probability of simultaneous version failure (as [10] imply) that is the basis for system reliability improvement. It is a more complex property, which may involve these simpler properties, whose exact composition is dependent upon the particular decision strategy employed. Put another way: useful software diversity is not a simple, context-free property of an N-version system.

However, empirical studies backed by theoretical analysis have claimed to show that N-version programming delivers a disappointingly low reliability enhancement, and why this might be so. The original motivation behind the multiversion idea was that independently developed versions would be likely to fail *independently*; "that is, faults in the different versions occur at random and are unrelated. Thus the probability of two or more versions failing on the same input is very small"[8] p. 96. However, the precise meaning of central concepts, such as 'independence', initially received little attention. Eckhardt and Lee [6], [7] began the task of formalizing some of these concepts and their interrelationships. An important outcome of their study was that *truly independently* prepared versions (where 'independence' is precisely defined) will necessarily fail dependently. This outcome is due to variation in difficulty of the inputs. If some inputs are harder than others (as is inevitably the case), then more failures are likely to occur on these inputs, no matter what version (or versions) are chosen.

Brilliant, Knight and Leveson[5] similarly work from the standpoint that "The amount of reliability improvement achieved [by majority voting in an N-version system] is *determined* by the degree of independence of the failures of the versions" (p. 238 citing [7] as the source — our emphasis). They analyse faults in their large study [8] of the Launch Interceptor (LI) problem, and conclude pessimistically: "we do not expect that changing

development tools or methods, or any other simple technique, would reduce significantly the incidence of correlated failures in N-version software" (p. 245). But the results of several experiments [4, 1, 2] do not provide support for this negative prognostication.

In this paper we develop an argument to reconcile these conflicting results. The crux of the matter is that the potential reliability of an N-version system is not always accurately estimated by means of either independence of failure or lack of coincident failure among the component versions.

The distribution of failures can be important as well as the property of whether versions fail in the same way, rather than whether they just fail on the same inputs. The diversity of distinct failures needs to be considered but has hitherto been largely unacknowledged and thus overlooked. Proper consideration of this property can raise dramatically the estimated N-version system reliability. Many of the earlier studies may therefore be seriously underestimating the potential reliability of the systems tested.

2 Littlewood & Miller model

Littlewood and Miller [10] developed a thorough conceptual model to underpin the multiversion approach to software engineering. In agreement with Eckhardt and Lee [7], the new model is surprising in that it relegates *independence* to a position of relative unimportance. Littlewood and Miller's model shows that, in theory, it is possible to produce better than independent failure behaviour: it is possible to have a multiversion system with negatively correlated failure.

At a more obviously practical level, they demonstrate that the limitations of dependent failure despite independent development can be overcome by adopting a policy of forced diversity at the methodology level. The use of diverse methodologies is shown to decrease the probability of simultaneous failure of several versions.

Two fundamental probabilities that derive from this model are: $E(\theta)$, the probability of a randomly chosen version failing on a randomly selected input, and $E(\theta^2)$, the probability of two randomly selected versions both

failing on a randomly selected input. These are considered critical indicators of the 'worth' of a multiversion system (insofar as the overall system can deliver a more reliable performance than its component versions) because the difference between $E(\theta)^2$ and $E(\theta^2)$ provides a measure of the degree of dependence of failure actually achieved.

The basis for computing these quantities is a score function

$$\nu(\pi, x) = \begin{cases} 1 & \text{if version } \pi \text{ fails on input } x \\ 0 & \text{if version } \pi \text{ does not fail on input } x \end{cases}$$

which can be tabulated as shown in Table 1 for an illustrative example in which 10 inputs are applied to 5 versions.

input	version number					row total
number	1	2	3	4	5	
1	1	1	1	1	1	5
2	1	0	1	1	0	3
3	0	0	0	0	0	0
4	1	1	1	1	1	5
5	1	0	0	0	0	1
6	1	1	1	1	1	5
7	1	0	1	1	1	4
8	0	0	0	0	0	0
9	1	0	0	1	1	3
10	0	0	0	0	0	0

Table 1: Tabulated scores for a 5-version system

Simplifying the model with the assumption that the N versions and the M inputs are both equiprobable, $S(.)$, the selection distribution over the possible programs, becomes $1/N$ for each program, and $Q(.)$, the usage distribution over the input space, becomes $1/M$ for each input, respectively. (These features, although important aspects of the model itself, do not bear on the argument to be developed.) The required coincident failure probabilities then become

$$E(\theta) = \frac{1}{N}\frac{1}{M}\sum_{\mathcal{P}}\sum_{\mathcal{X}}\nu(\pi, x)$$

and

$$E(\theta^2) = \frac{1}{M}\sum_{\mathcal{X}}(\sum_{\mathcal{P}}\nu(\pi, x)\frac{1}{N})^2,$$

where the summations are over both inputs ($\mathcal{X}$) and versions ($\mathcal{P}$).

In addition, Littlewood and Miller note that "the degree of departure from independent behaviour is governed by $Var(\theta)$", where this variance, $Var(\theta) = E(\theta^2) - E(\theta)^2$ (p. 1599).

Using the data in the score table, we obtain the sample estimates $E(\theta) = 0.52$ and $E(\theta^2) = 0.44$, but as $E(\theta)^2 = 0.27$ we can see that the five versions do not exhibit independence with respect to failure behaviour. The naive calculation assuming independence (i.e. $E(\theta)^2$) underestimates the the actual two-version failure by a factor of 1.6 and $Var(\theta) = 0.17$. When $Var(\theta) = 0.0$ we have independent failure behaviour.

The score function information can be more succinctly captured in a 'coincident failure' table, i.e. a tabulation of the number of occasions on which exactly n versions fail on an input. The coincident failures from the score tabulation are given in Table 2, in which $p_n = \frac{the\ number\ of\ times\ exactly\ n\ versions\ fail}{the\ total\ number\ of\ inputs}$

n	no. of times n versions fail	p_n
0	3	0.3
1	1	0.1
2	0	0.0
3	2	0.2
4	1	0.1
5	3	0.3

Table 2: the coincident-failure table for the 5-version system

and is the (estimated) probability that exactly n versions will fail coincidently on a random input, where $0 \leq n \leq N$.

Using p_n values alternative forms for the two failure probabilities are obtained.

$$E(\theta) = \sum_{n=0}^{N} \frac{n}{N} p_n$$

and

$$E(\theta^2) = \sum_{n=0}^{N} \frac{n^2}{N^2} p_n$$

Notice that $E(\theta^2) \geq E(\theta)^2$ and that equality only occurs when a $p_n =$

1.0, say p_{n_1}, for then both summations contain a single non-zero term:

$$E(\theta) = \frac{n_1}{N} p_{n_1} \text{ and } E(\theta^2) = \frac{n_1^2}{N^2} p_{n_1}$$

and as $p_{n_1} = 1.0$

$$E(\theta)^2 = E(\theta^2).$$

Littlewood and Miller further extend their model to encompass selection of versions from two sets of versions, A and B. A major result is that when selecting two versions at random, but one from each set, it is possible to get better than independent behaviour!

With $E(\theta_A \theta_B)$ denoting the probability that two versions both fail on a random input when selected at random one from set A and one from set B, and $E(\theta_A)$ and $E(\theta_B)$ denoting $E(\theta)$ for each set separately, the essential equation is:

$$E(\theta_A \theta_B) = Cov(\theta_A, \theta_B) + E(\theta_A).E(\theta_B)$$

where the covariance, $Cov(\theta_A, \theta_B)$, governs the degree of departure from independence when the two versions are chosen from different sets.

A point they make is that it is possible that $Cov(\theta_A, \theta_B) < 0.0$, which means that, in principle, better than independent failure behaviour can be achieved when selecting from two different sets. ¿From the earlier argument (i.e. that $E(\theta^2) \geq E(\theta)^2$)) it is clear that the single-set 'equivalent', $Var(\theta)$, has a minimum bound of zero, i.e. independent failure behaviour is the best that can be expected when using a single set of versions, but not when using two (or more) sets.

However, for practical purposes (e.g. estimating coincident failure probabilities for an actual N-version system) this model requires some modification, and by so doing we can also remove the odd restriction on single-set systems. Necessarily finite input test spaces and small finite numbers of versions need to be accommodated in any practical application.

3 The Krzanowski & Partridge model

Krzanowski and Partridge [9] have provided a development of the Littlewood and Miller model which makes no pretence to emulate the completeness of

the precursor conceptual model, but concentrates instead on a framework for practical application. Notice that the expression for $E(\theta^2)$ assumes that version sampling is with replacement which means, for example, that the probability of exactly 1 version failing contributes in a small, but positive way, to the probability that two versions will both fail (because with-replacement sampling will allow the same version to be selected twice). It would also mean that in a multiversion system in which every failure is unique (i.e. no coincident failure) the probability of two-version failure is not zero.

The Littlewood and Miller theory assumes very large or infinite populations of program versions and inputs. Under such assumptions sampling with replacement and sampling without replacement are virtually indistinguishable. But in any practical application these two sampling techniques will deliver significantly different results, and sampling without replacement would appear to be the preferred option as we are interested in the extra reliability to be gained from use of two (or more) different versions.

As a result $p(1)$, the probability that a randomly chosen version will fail on a randomly chosen input, is estimated in exactly the same way as $E(\theta)$, but $p(2)$, the probability that two randomly chosen versions will both fail on a randomly chosen input is now

$$p(2) = \sum_{n=0}^{N} \frac{n}{N} \frac{(n-1)}{(N-1)} p_n$$

instead of $E(\theta^2)$. Notice that $p(2) < E(\theta^2)$ for all non-zero $p(2)$, because $\frac{n(n-1)}{N(N-1)} < \frac{n^2}{N^2}$ for $n < N$.

The new measure of dependence of behaviour is $p(2) - p(1)^2$, and no longer has a lower bound of zero. The single-set model now parallels the multiset model.

The relationship between the important probability estimates is:

$$p(2) < E(\theta^2) \geq E(\theta)^2 = p(1)^2$$

Within this model a coincident-failure diversity (CFD) measure has been developed.

$$CFD = \frac{1}{1 - p_0} \sum_{n=1}^{N} \frac{(N-n)}{(N-1)} p_n$$

if $p_0 < 1.0$, and $CFD = 0.0$ when $p_0 = 1.0$.

This diversity measure has a minimum value of 0.0 when there are no version failures (i.e. $p_0 = 1.0$) and when all version failures are coincidental over all N versions (i.e. $p_0 + p_N = 1.0$). It has a maximum value of 1.0 when all version failures are unique (i.e. $p_0 + p_1 = 1.0$).

Using the coincident failure table given earlier, Table 2, we obtain $p(2) = 0.42$ (cf $E(\theta^2) = 0.44$) and $CFD = 0.32$.

If m_n is the number of tests that fail on n versions, an alternative formulation of CFD can be obtained from

$$\begin{aligned}
\frac{p_n}{1-p_0} &= \frac{m_n/M}{1-m_0/M} \\
&= \frac{m_n}{M-m_0} \\
&= \frac{m_n}{m_1+m_2+\ldots+m_N} \\
&= \frac{the\ number\ of\ tests\ that\ failed\ on\ n\ versions}{the\ number\ of\ tests\ that\ failed\ on\ at\ least\ one\ version} \\
&= f_n
\end{aligned}$$

where f_n is the (conditional) probability that a test failure will fail on exactly n versions.

Hence CFD can be written

$$CFD = \sum_{n=1}^{N} \frac{(N-n)}{(N-1)} f_n$$

This diversity measure, which supercedes an earlier one [11], is (under certain conditions) a useful indicator of the potential for reliability enhancement in a multiversion system under majority voting strategies. The model [9] also generalizes this measure to the multiset case such that the relative potential of alternative system structures (such as a diverse 2-out-of-3 system or a homogeneous 2-out-of-3 system) can also be effectively assessed. As Littlewood and Miller's analysis shows "the diverse design is not always superior to the homogeneous design"(p. 1605). The multiset diversity indicators permit assessment of the relative merits of these two options which have, in fact, been examined [12].

4 Distinct failure diversity

In all of the previous work on multiversion software, the focus has been on version success and version failure with an emergent concentration upon minimizing the likelihood of coincident failure between any two versions. However, if the problem implemented admits the classification of version outputs into distinct categories, and the number of such categories is greater than two, then a further 'dimension' of software diversity opens up. Two versions may fail coincidently but distinctly, and this aspect of diversity may be exploited to enhance system reliability. Most significantly, this distinct-failure diversity (DFD) may not just add to coincident-failure diversity, it can be a system reliability supporting feature in its own right, *even when the N-versions exhibit no coincident-failure diversity*! It is possible to obtain maximum system reliability through exploitation of DFD despite total lack of coincident-failure diversity.

Consider a five-version system in which all inputs result in three versions failing and only two succeeding. In this system, the majority of versions fail coincidently on all inputs, and it would seem to be an extremely poor system. For if the failures always coincide with the same wrong result, a majority-vote strategy will always deliver an incorrect outcome, a system of maximum unreliability. But if the failures on any input are always distinct, a majority-in-agreement strategy would always yield a correct outcome, a 100% reliable system.

These are clearly extreme, and perhaps unlikely, possibilities, but they do, nevertheless, underline the point that multiversion system performance is not governed by the coincident failure characteristic captured in either $E(\theta^2)$ or $p(2)$. An important aspect of diversity has been overlooked. All of the previous measures will provide the same results on a given system test, whatever the distribution of failures among the distinct alternatives. But any comprehensive treatment must consider the distribution of failures among the set of *distinct* wrong answers. We need a notion of distinct-failure diversity (DFD). Consideration of the distribution of failures among

the set of distinct alternatives adds a further dimension to the problem (and potential) of multiversion software. In addition to scoring success or failure, the category of failure must be taken into account.

We assume that the problem permits classification of each input into one of $c+1$ categories, and thus that an error can be manifest as one of c distinct wrong categorizations (because, for any input, one of the output categories is correct), where $c \geq 1$.

For example, in a 'letter recognition' problem each input derives from a distinct letter of the alphabet. There are thus $(c+1) = 26$ output categories of which 25 are wrong. (However, the 'correct' category will vary from input to input, in general it is a different one of the 26 for each distinct input letter.)

Returning to the earlier illustrative example of 10 inputs on 5 versions, the extra information is obtained by decomposing the failure totals (which recorded the number of versions that failed on each input) into the actual category of each output. If the problem output can be classified into 6 distinct categories, a possible result is given in table 3.

input	output category						correct	total
number	1	2	3	4	5	6	category	failures
1	0	**3**	**1**	0	0	**1**	1	5
2	**1**	2	**2**	0	0	0	2	3
3	0	0	5	0	0	0	3	0
4	**2**	**1**	0	0	**2**	0	4	5
5	**1**	0	0	0	4	0	5	1
6	0	0	**5**	0	0	0	6	5
7	0	**4**	0	1	0	0	4	4
8	0	5	0	0	0	0	2	0
9	**1**	2	0	**1**	**1**	0	2	3
10	5	0	0	0	0	0	1	0

Table 3: An incidence matrix for the 5-version system

The previous row totals have been preserved, but the individual entries record the number of incidences in each category of ouput observed on each input. This is no longer a score function tabulation (the columns now exhibit

occurrences of specific category outputs rather than version failures). It is an incidence matrix. Failures are shown in bold in this matrix.

We define the (estimated) probability that exactly n versions will fail identically on a random test, t_n.

$$t_n = \frac{\text{number of times that n versions fail identically}}{\text{total number of distinct input failures}}$$

where the denominator is the number of bold non-zero entries in the incidence matrix, and the numerator is obtained by counting the number of bold entries for each value of n.

Thus for the incidence matrix illustrated (Table 3) there are fourteen distinct (non-zero) failure categories, eight of which have just one failure, three have two failures, one has three failures, one has four and one has five. Thus we obtain: $t_1 = \frac{8}{14} = 0.57$; $t_2 = \frac{3}{14} = 0.21$; $t_3 = \frac{1}{14} = 0.07$; $t_4 = \frac{1}{14} = 0.07$; $t_5 = \frac{1}{14} = 0.07$.

Using this probability estimate DFD is defined:

$$DFD = \sum_{n=1}^{N} \frac{(N-n)}{(N-1)} t_n$$

if $(total\ number\ of\ identical\ test\ failures) > 0$
otherwise $DFD = 0$.

Applying this measure to the data in the incidence matrix, we obtain

$$DFD = 0.57 + 3/4 \times 0.21 + 2/4 \times 0.07 + 1/4 \times 0.07 = 0.79$$

This diversity measure has another minimum of zero when $t_N = 1.0$, i.e. all versions fail coincidently on every test case. DFD has a maximum value of 1 when $t_1 = 1.0$, i.e. every version failure is unique. However, if $c \geq N$ then all versions may fail uniquely on every test case, and there is no overall diversity, merely a maximum diversity within the set of (total) failures. But in this case $CFD = 0$, because $p_N = 1.0$. CFD measures the overall diversity of failure (without regard to the distribution of failures within the set of distinct possibilities). This observation suggests that an overall diversity measure must take both types of diversity into account.

In order to illustrate the significance of diversity among specific failures in contrast to the usual concept of coincidence of failure between versions, we can consider a variation of the previous illustrative example.

We have 10 inputs, 5 versions, and 6 distinct output categories. Suppose exactly 3 versions fail on all inputs, but not the same three versions and the failures are in three distinct categories, on each occasion. We then have no individual version that is 100% correct:

$$p_0 = 0.0; p_1 = 0.0; p_2 = 0.0; p_3 = 1.0; p_4 = 0.0; p_5 = 0.0$$

and

$$t_1 = 1.0; t_2 = 0.0; t_3 = 0.0; t_4 = 0.0; t_5 = 0.0$$

¿From these basic probability estimates we can compute

$$\begin{aligned} p(1) &= 3/5 \times 1.0 = 0.60 \\ p(2) &= 3/5 \times 2/4 \times 1.0 = 6/20 = 0.30 \\ E(\theta^2) &= 9/25 = 0.36 \\ CFD &= 2/4 \times 1.0 = 0.50 \\ DFD &= 4/4 \times 1.0 = 1.0 \end{aligned}$$

The coincident failure diversity is at the midpoint of its range, but the distinct failure diversity is a maximum. Notice that because a majority of versions always fail (by assumption), a majority can never deliver a correct answer. But a majority-in-agreement strategy will always deliver a correct answer.

This particular system does exhibit some coincident-failure diversity, but as the size of the version set increases, the coincident-failure diversity decreases. Some examples that reveal this trend are given in Table 4. In all cases it is assumed that $(N-2)$ versions fail uniquely on every input, which means that $c \geq (N-2)$.

As Table 4 clearly shows, under conditions of extreme DFD, which supports total system reliability, the coincident-failure diversity approaches zero and probability of two-version failure approaches unity as the number of versions increases.

In addition, independence of failure, measured in terms of the agreement between $E(\theta^2)$ and $E(\theta)^2$ (as [10] propose), is achieved in all sets! This is because, under this particular distribution of failure, both quantities reduce to $(N-2)^2/N^2$. In general, as any $p_n \to 1.0$, $E(\theta)^2 \to E(\theta^2)$, and hence independence of failure behaviour as indicated by this measure.

But if independence of failure is measured in terms of the agreement between $p(2)$ and $p(1)^2$, we can see that with small values of N the sets exhibit better than independent failure behaviour (i.e., $p(2) < p(1)^2$) which gradually deteriorates to mere independence at set size 50. Thus independence of failure (in these cases) is also no guide to potential reliability.

Distinct-failure diversity is thus a significant characteristic of a version set, in the sense that it may indicate potential for system reliability enhancement when coincident-failure diversity does not. But because distinct-failure diversity alone is not a good indicator of system reliability potential (e.g. $DFD = 1$ and yet all versions fail on all tests), measures of multiversion system diversity should include components from both types of diversity. It might thus be more useful to integrate consideration of these two forms of diversity from the outset.

4.1 An integrated approach

Suppose that there are $c+1$ output categories (and thus c distinct failure categories for each input test), and that n versions fail on a particular test. We now consider just the c 'failure' categories for each test.

We divide the possibilities into four cases:

(a) when there are fewer failures than failure categories, i.e., $n < c$;

(b) when there are the same number of failures as failure categories, i.e., $n = c$;

(c) when the number of failures is an exact multiple of the number of failure categories, i.e., $n = mc$ for integer m;

(d) when there are more failures than failure categories, but the former is

not an exact multiple of the latter, i.e., $n = mc + j$ for integer m and j.

The 'most diverse' distribution of failures is when they are as evenly spread as possible among the categories. The four cases become:

(a) if $n < c$, then there are n each with 1 failure and $(c - n)$ categories without any failures; pictorially we have, $\underbrace{1, 1, 1, \ldots, 1}_{n}\underbrace{0, \ldots, 0}_{c-n}$

(b) if $n = c$, then each of the categories has 1 failure; pictorially, $\underbrace{1, 1, 1, \ldots, 1}_{c}$

(c) if $n = mc$ for integer m, then each of the categories has m failures; pictorially, $\underbrace{m, m, m, \ldots, m}_{c}$

(d) if $n = mc + j$ for integer j and m, then there are j categories each with $(m+1)$ failures, and $(c-j)$ categories each with m failures; pictorially, $\underbrace{m+1, m+1, m+1, \ldots, m+1}_{j}\underbrace{m, \ldots, m}_{c-j}$

All four cases can be expressed in the form $n = mc + j$ for suitable (i.e. integer or zero) j and m.

Thus in general the 'most diverse' frequency distribution, when expressed as a proportion of n, has the form:
$\underbrace{\frac{m+1}{n}, \frac{m+1}{n}, \ldots, \frac{m+1}{n}}_{j}\underbrace{\frac{m}{n}, \ldots, \frac{m}{n}}_{c-j}$ for some j $(0 \leq j < c)$

One way of measuring distinct-failure diversity of a row (i.e. test) is therefore to calculate its 'similarity' to the above 'most diverse' situation.

Let us write $a_1, a_2, \ldots, a_c$ as the proportions in the failure categories of the row ($\sum a_i = 1$), and $\frac{m+1}{n}, \ldots, \frac{m+1}{n}, \frac{m}{n}, \ldots, \frac{m}{n}$ as the proportions in the 'most diverse' case.

These are both just frequency distributions, and a useful measure of similarity (or 'affinity') between them [3] is

$$d = \sum_{i=1}^{j} \sqrt{a_i \times \frac{m+1}{n}} + \sum_{j+1}^{c} \sqrt{a_i \times \frac{m}{n}}$$

Providing the a_i are arranged appropriately, the 'most diverse' frequency distribution will always have the value $d_{max} = 1$, while the 'least diverse' $(0, \ldots, 0, \frac{n}{n})$ will always have value $d_{min} = \sqrt{\frac{m}{n}}$.

We now define overall diversity, OD, as a *weighted* version of the previous definition of coincident-failure diversity, CFD:

$$OD = \sum_{n=1}^{N} \left(\frac{N-n}{N-1} \right) f_n w_n$$

where w_j is the average of the d values for all rows with j failures.

Notice that $w_i = 1$ for all i when there is only one category of failure. So OD just becomes CFD in this case (as is required).

So for the failure results illustrated in Table 3, $w_1 = 1, w_2 = 0, w_3 = \frac{1}{2}(0.804 + 1.0) = 0.902, w_4 = 0.5$ and $w_5 = \frac{1}{3}(0.745 + 0.766 + 0.447) = 0.653$ (see Appendix A for detailed calculation of the d values) and

$$OD = \frac{1}{6} \times 1 + \left(\frac{N-3}{N-1} \right) \times \frac{1}{6} \times 0.902 + \left(\frac{N-4}{N-1} \right) \times \frac{1}{6} \times 0.5 + \left(\frac{N-5}{N-1} \right) \times \frac{3}{6} \times 0.653.$$

When the value for N is substituted, we thus obtain

$$OD = \frac{1}{6} + \frac{2}{4} \times \frac{1}{6} \times 0.902 + \frac{1}{4} \times \frac{1}{6} \times 0.5 = 0.26$$

whereas

$$CFD = 0.32 \text{ and } DFD = 0.79.$$

As these figures illustrate, for the example chosen, the overall diversity is not very high, but most of it derives from diversity among failure categories rather than from lack of coincident failures.

5 The impact of distinct-failure diversity

Exploitation of distinct-failure diversity rests on the property that, within a version set, a few correct results (as few as two) can be recognized as such in contrast to a collection of different wrong results. In the case of a simple majority-voting strategy for generating the multiversion system outcome, the potential benefit of distinct-failure diversity can be quantified and illustrated.

For a multiversion system, containing N versions each of which is designed to compute a function with $c+1$ distinct output categories for each input, we can apply M tests and obtain a coincident-failure table as illustrated above. From this table we obtain the p_n values, each is the probability that exactly n versions will fail on a random input.

Then the probability that a majority vote will be correct, p_{maj}, is

$$p_{maj} = \sum_{n=0}^{k} p_n$$

where k is the largest minority number, i.e. $k = Ndiv2$[1].

This probability is based solely on the coincident-failure concept, for if coincident failure is minimized then the probability of large numbers of versions failing coincidently will be similarly minimized. Hence, all p_n, for $n > k$ will be minimized, and so p_{maj} will be maximized. In fact, the probability of n versions failing coincidently, can be arbitrarily high for all $n \leq k$, and still the system will deliver a 100% majority-vote performance. But any coincidence of failure in greater than k versions will undermine the majority vote.

So, even from a coincident-failure perspective, it is not the minimization of coincident failure between any two versions that is required (assuming a majority-vote treatment). It is less demandingly, the restriction of coincident failure to no more than k versions. The required characteristic is a threshold feature, dependent upon the total number of versions N, not a continuum within which a minimum is required and the smaller the better.

Once the multiversion system strategy includes the property of distinguishing between different errors (i.e. a majority-in-agreement vote), then the potential of distinct-failure diversity can also be exploited. Where is this additional potential reliability to be found? It is in the p_n values from p_{k+1} to p_{N-2}. It is easy to appreciate this potential from a consideration of the two extremes of distinct-failure diversity:

no distinct-failure diversity : In this case all failures are identical, there is no way to distinguish one failure from another and so we are ef-

[1] we assume for simplicity that N is odd

fectively limited to coincident-failure diversity. If $(k+1)$ (or more) versions all fail, then the majority in agreement is precisely the same as a success/failure majority vote.

maximum distinct-failure diversity : In this case every failure is unique, so whenever there are at least two correct versions, the majority in agreement will be correct. The limit on this, with respect to p_n values, is that at least two versions must be correct. Hence, only the simultaneous failure of either all N or $(N-1)$ versions will detract from the overall system performance — i.e. p_{N-2} is the limiting probability of simultaneous failure that will not adversely effect system reliability.

We thus define the probability that a majority in agreement will be correct, p_{mia}, as

$$p_{mia} = \sum_{n=0}^{k} p_n + \sum_{n=k+1}^{N-2} p_n g_n$$

where g_n is the conditional probability that $(N-n)$ 'non-failure' versions are in the majority given n failures distributed over c failure categories considered category by category, i.e., $(N-n) > max_i(n_i)$, where n_i is the number of versions that fail in category i (so $n = \sum_{i=1}^{c} n_i$).

Suppose that f_i is the (conditional) probability that a failure falls in category i. Then the probability that n failures are distributed among the c categories with n_i in category i is

$$\frac{n!}{n_1! n_2! \ldots n_c!} f_1^{n_1} f_2^{n_2} \ldots f_c^{n_c}$$

Now the $(N-n)$ non-failures will be in the majority if $(N-n) > max_i(n_i)$ and there will be various partitions $n_1, \ldots, n_c$ of n which are consistent with this inequality in general. Let $\mathcal{D}(n)$ be the set of such partitions.

Then

$$g_n = \sum_{\mathcal{D}(n)} \frac{n!}{n_1! \ldots n_c!} f_1^{n_1} f_2^{n_2} \ldots f_c^{n_c}$$

and

$$p_{mia} = \sum_{n=0}^{k} p_n + \sum_{n=k+1}^{N-2} \sum_{\mathcal{D}(n)} p_n \frac{n!}{n_1! \ldots n_c!} f_1^{n_1} f_2^{n_2} \ldots f_c^{n_c}$$

If we assume that a failure falls at random into one of the c categories, then $f_1 = f_2 = \ldots = f_c = \frac{1}{c}$ and the expression simplifies to

$$p_{mia} = \sum_{n=0}^{k} p_n + \sum_{n=k+1}^{N-2} \sum_{\mathcal{D}(n)} p_n \frac{1}{c^n} \frac{n!}{n_1! \ldots n_c!}$$

6 Empirical studies

We illustrate the significance of distinct-failure diversity by drawing on results from two, rather different, previously published studies of multiversion software systems.

6.1 Previous results on the LI problem

As part of an empirical study of the Littlewood and Miller model, Adams and Taha [1] used the LI problem (specified in, and previously studied by Knight and Leveson [8,5]). They arranged for the independent development of implementations of this problem using two implementation target languages, Prolog and Modula2. Each version has to pass an acceptance test of 200 randomly generated test cases (a different random 200 for each acceptance test) before it was accepted as an experimental version. They obtained eleven versions, five Prolog versions and six Modula2 versions.

Using a 'gold' program (a supposedly correct implementation of the LI problem) a test set of 10,000 test cases was generated, but 122 of these tests caused an 'exception condition' when executed by the gold program. All of these unsatisfactory tests were traced to ambiguity in the specification. The crux of the ambiguity is whether certain input parameter values are completely determined by the declarations given, or if additional restrictions implied by details in the functional specification should be considered. In the absence of any further information they applied the two test sets (i.e. the original 10,000 and the 9878 obtained by omitting the problematic 122 cases) to the set of six Modula2 versions. The test results are given in Table 5, from which we can determine that the average performance of the individual versions (i.e. average "prob(succ)" which is computed as

$\frac{the\ number\ of\ tests\ computed\ correctly}{total\ number\ of\ tests,\ M}$) is 98.98% when $M = 10000$ and 99.31% when $M = 9878$.

The consequent coincident-failure results are given in Table 6.

As the results in the two Modula2 tests were not markedly different, and taking the view that the specification ambiguities effectively invalidated the 122 'problematic' tests, the study was completed using the smaller test set, i.e. $M = 9878$. Table 7 gives the results for the set of five Prolog versions.

The major result from this study was that when the Modula2 and Prolog versions were treated as disjoint sets, each derived from a rather different methodology (as enforced by the radically different target languages), the probability of coincident failure of two versions selected at random but one from each set, $E(\theta_{Mod}\theta_{Pro})$, was at the level that independence of failure would produce. This was confirmed by the value for Littlewood and Miller's error correlation coefficient, which was essentially zero (actually computed as -3.633×10^{-4}). Adams and Taha conclude: "In essence, the positive effect of diverse methodologies has compensated for the negative effect of input data variance."([1] p. 89)

Considering only the 9878 test set, and using "Mod" to denote the set of six Modula2 versions and "Pro" to denote the five Prolog versions, the essential quantities were computed to be:

$$\begin{aligned} p(2)_{Mod} &= 1.869 \times 10^{-3} \\ E(\theta^2_{Mod}) &= 2.699 \times 10^{-3} \\ E(\theta_{Mod})^2 &= 4.692 \times 10^{-5} \end{aligned}$$

$$\begin{aligned} p(2)_{Pro} &= 7.187 \times 10^{-4} \\ E(\theta^2_{Pro}) &= 1.397 \times 10^{-3} \\ E(\theta_{Pro})^2 &= 1.255 \times 10^{-5} \end{aligned}$$

$$\begin{aligned} E(\theta_{Mod}\theta_{Pro}) &= 2.357 \times 10^{-5} \\ E(\theta_{Mod}) \times E(\theta_{Pro}) &= (6.850 \times 10^{-3}) \times (3.543 \times 10^{-3}) = 2.427 \times 10^{-5} \end{aligned}$$

It is the similarity of the last two values that indicates independence of failure between two versions when selected one from each set. And it is the discrepancy between the second and third, and between the fifth and sixth (two orders of magnitude in both cases) that indicates lack of independent failure behaviour within each set of versions. Notice also that use of the (more accurate) $p(2)$ values, which we have added, reduces the apparent lack of independent failure behaviour.

Using the above data, and applying our model [9], estimates of the reliability of multiversion systems based on the eleven versions can be computed. Table 8 gives some informative values. Although the coincident-failure diversity values are high (both > 0.9) the individual versions are so reliable that the scope for majority-vote improvement is less than 0.2% (over the most reliable individual versions). In the Modula2 set the majority-vote performance is as good as the best individual version (and approximately 0.5% better than the average which amounts to removal of two thirds of the residual error), and in the Prolog set the five-version system is slightly better than the best individual version. What is not revealed by these results though is that a collective outcome (such as majority voting) from an N-version system is typically a more secure estimate of future performance than a similar result from a single version because diverse sets of versions are less sensitive to specific features of the actual test set.

Inspection of the coincident-failure tables also reveals that there is little scope for extra reliability from distinct-failure diversity. Firstly, because there are so few versions (e.g. when $N = 5$, $k + 1 = 3$ and $N - 2 = 3$ so only the instances of exactly three versions failing can be exploited using distinct-failure diversity). Secondly, because of the actual failure profiles exhibited by the two sets of versions (i.e. a minimum of failures in the potentially exploitable range and peaks at the irretrievable maximums, when N versions all fail).

Nevertheless, some potential gain from distinct-failure diversity is evident. When $M = 10000$ (table 6) there are six Modula2 versions so $k+1 = 3$

and $N - 2 = 4$. This means that when exactly three and four versions fail the majority vote will not be correct, but majority-in-agreement will be, *provided the failures are all distinct.* If we have maximum distinct-failure diversity in these two cases (i.e. $g_3 = g_4 = 1.0$) then the system performance will be an improvement of 0.18% (i.e. $p_3 + p_4$) over the simple majority-vote strategy. This is a further 56% reduction in the residual error — majority vote ($p_0 + p_1 + p_2$) of 99.68% increases to majority-in-agreement of 99.86%.

Using the Prolog results (Table 7) we can illustrate the lack of necessary connection between independence of failure and coincidence of failure probabilities with system performance.

If we collapse the first three p_n values (i.e. set p_2 to $p_0 + p_1 + p_2 = 0.999088$, and set $p_0 = p_1 = 0.0$), we have drastically changed the distribution of failures in this set. According to such a distribution, two of the five versions fail on almost every test, but the system reliability under a majority-vote strategy remains unaltered.

However, this is far from true about the supposed indicative estimates:

$$p(2)_{Pro} = .1006 \;\; E(\theta^2_{Pro}) = .1606 \;\; E(\theta_{Pro})^2 = .1604 \;\; CFD = 0.750$$

The Littlewood and Miller measure ($E(\theta^2_{Pro}) - E(\theta_{Pro})^2$) indicates that the set now exhibits independence of failure behaviour, and our measure ($p(2)_{Pro} - E(\theta_{Pro})^2$) indicates much better than independent failure behaviour. Yet the set, as a whole, still exhibits precisely the same reliability under a majority-vote decision strategy.

Notice that the CFD value decreases significantly, but is still not dramatically misleading (especially as it is not a majority-vote indicator, but a general diversity measure, and there is less diversity in the new system). And if we collapse only $p_0 + p_1$ to $p_1 = 0.997671$ and $p_0 = 0.0$, then CFD changes little — from 0.913 to 0.997 – but the probability estimates again indicate independence of behaviour, or better. The estimate $p(2)_{Pro}$ barely changes but $E(\theta^2_{Pro})$ increases by an order of magnitude. This latter result might be taken to indicate that coincident failure is much higher in the altered set. Consequently, diversity is lower (as per [10], p. 1609) and thus system

reliability should be much worse. But, as we know, under majority-vote, at least, nothing has changed concerning overall system reliability.

6.2 Inductive programming and N-version systems

In a previous study [13] an N-version system was constructed to classify images as one of the 26 upper case letters, A to Z. The versions were neural networks, each differently trained on example data, to produce a nine-version system. In fact, two such systems were constructed using two types of neural network, Multilayer perceptrons (MLP) and Radial basis function (RBF) nets which provided a major source of methodological diversity in the N-version systems.

It is a difficult classification problem because the original letter images were assembled from many different fonts (including, for example, italic and gothic) and then randomly distorted both horizontally and vertically. Some of the resultant images are extremely difficult, probably impossible, to classify correctly. Each image was then transformed into a vector of 16 numerical characteristics, and a randomised file of 20,000 such feature vectors (each paired with the correct classification, and publically available at aha@ics.uci.edu) forms the data that defines the problem. Following previous attempts to produce optimal implementations of this letter recognition task, the first 16,000 data vectors are available for developing the implementation, and the final 4,000 constitute the system test.

In Table 9 we present the failure results for two nine-version neural-net systems that were developed. The first, $pick_{CFD}$, was chosen (from a larger pool of versions) with the objective of maximising the CFD value, and the second was similarly chosen but to maximize the DFD value. The two systems contained six versions in common which included the best and worst individual versions. The remaining three versions must therefore be responsible for the differences in diversity that are recorded.

Within this table the coincident-failure probabilities that can provide extra reliability for a majority-in-agreement strategy have been emphasized. It is the p_n values from $k+1$ to $N-2$, and when $N=9$ this is p_5, p_6 and p_7.

Performance and diversity indicator values for these two systems are given in Table 10.

Both systems illustrate the extra reliability gain obtained by exploiting distinct-failure diversity (cf p_{maj} with p_{mia}) — over 4% and nearly 7%, a 30% reduction in residual error. The larger gain is obtained from the system with the larger DFD value.

Notice that the larger DFD in the $pick_{DFD}$ system nearly compensates for the considerably poorer performance of the three 'variable' versions it contains (the system average is 10% lower than for $pick_{CFD}$). This last observation suggests that the production and exploitation of a small increase in distinct-failure diversity can compensate for individually 'weak' versions in a system. Distinct-failure diversity is thus an important feature of N-version systems at a wholly practical level.

Using the model developed earlier, we can compute the actual g values that would be obtained if failures fell into categories at random (see Appendix B). The calculated values are $g_5 = 0.9997$, $g_6 = 0.9708$ and $g_7 = 0.3969$. This random distribution of failures with its characteristic decrease in g_n as n increases is not close to a maximally (distinct-failure) diverse result. In order to obtain an average g_n value of about 0.9, c needs to be an order of magnitude greater than N. This may be a useful result for neural-computing technology which should be amenable to strategies involving 'engineered randomness'.

7 Discussion and conclusions

In previous studies, both empirical and analytic, to explore and understand the potential inherent in the N-version approach to software reliability, the emphasis on achievement of independent failure behaviour or on minimization of coincident failure is both misguided and incomplete. We have shown that neither by itself (nor together) is a guide to the potential system reliability enhancement that a move to N-versions will provide. Minimization of the crucial probabilities, either $E(\theta^2)$ or $p(2)$, is unnecessarily demanding because the relationship between such measures and majority-vote reliability

in an N-version system is not one of inverse proportionality. It is discontinuous, quite markedly so. It is a 'threshold' phenomenon: there is no adverse effect of coincident failure for any p_n values when $n = 1, \ldots, k$, an inverse proportionality relationship comes into play only when more than a minority of versions fail simultaneously (i.e. for p_n values when $n = k+1, \ldots, N$). It is not the probability of 2-version failure that is important so much as the distribution of non-zero p_n values with respect to the minority number, k. The coincident-failure diversity measure, CFD, is designed as a general measure of coincident-failure diversity (i.e. not simply an indicator of majority-voting potential in an N-version system) within which p_n values are weighted inversely to n in order to approximate this discontinuity.

The recognition and exploitation of distinct-failure diversity would appear to be yet another unacknowledged, but potentially productive, viewpoint to adopt. It switches the emphasis on the concept of common failure from one of failure on identical inputs, to one of identical wrong outputs. It thus opens up new possibilities for forced diversity in that the demand is not that alternative versions no longer fail on the same inputs, but that they no longer fail in precisely the same ways. This latter requirement would seem to be somewhat easier to satisfy, although it is currently unclear how it can be met.

It fits well as a natural expansion of the fault-tolerant multiversion strategy to distribute the inevitable error in order to minimize its adverse effects on the overall system. The sum of previous work has been to stress the distribution of error, viewed as failure (the undifferentiated alternative to success), such that component versions do not fail coincidently, i.e., do not make errors on the same inputs. The current work moves the focus of concern to the distribution of specific errors among the set of distinct failure possibilities which can result from each particular input. While it is, of course, necessary to obtain sufficient successes within the version set, it is also permissable to have coincident failures provided they are distinct. It relegates lack of coincident failure to be merely one, non-definitive, indicator of system reliability potential.

Curiously, one earlier study [8] appears to indicate the importance of distinct failures when the authors write that "the basic assumption [is] that the probability of common mode failure (*identical incorrect output given the same input*) is very low" (p. 97, our emphasis). But they then test for, and base their conclusions on notions of lack of independence or coincident failure (i.e. failure in the same input) only. And a follow-up study focuses exclusively on the minimization of coincident failure [5], "which means that both versions failed and includes failures with both identical and non-identical outputs"(p. 240).

Taking a broader view, this work diverts the quest for reliable software even further away from the dominant path which aims for total error elimination. It offers a richer infrastructure for the notion of software error such that the scope for devising fault-tolerant strategies is widened. Or from a software production viewpoint: it suggests that there are more options for the design of software engineering techniques to distribute the inevitable errors harmlessly, rather than to achieve total error elimination. The latter option may be philosophically more satisfying, but the former is more pragmatically sound in that every non-trivial software system will contain errors.

8 Acknowledgements

William B. Yates is thanked for constructing the two nine-version systems, and Phillis Jones for her work on implementing the model, both were supported by a grant from the EPSRC/DTI Safety-Critical Systems Programme (GR/H85427) whose support is gratefully acknowledged. J Mack Adams is thanked for his critical reading of earlier drafts.

9 References

1. J. M. Adams and A. Taha, "An experiment in software redundancy with diverse methodologies", *Proc. 25th Hawaii Internat. Conf. on System Sciences*, January, pp. 83-90, 1992.

2. A. Avižienis, M. R. Lyu and W. Schütz, "In search of effective diversity: a six-language study of fault-tolerant flight control software", *Dig. 18th Ann. Int. Symp. Fault-Tolerant Computing*, June, pp. 15-22, 1988.

3. A. Bhattacharyya, "On a measure of divergence between two multinomial populations," *Sankhya A*, vol. 7, pp. 401-406, 1946.

4. P. G. Bishop, D. G. Esp, M. Barnes, P. Humphreys, G. Dahll and J. Lahti, "PODS — A project on diverse software," *IEEE Trans. on Software Eng.*, vol. SE-12, no. 9, pp. 929-940, 1986.

5. S. S. Brilliant, J. C. Knight and N. G. Leveson, "Analysis of faults in an N-version software experiment," *IEEE Trans. on Software Eng.*, vol. 16, no. 2, pp. 238-247, 1990.

6. D. E. Eckhardt and L. D. Lee, "A theoretical basis for the analysis of redundant software subject to coincident errors," NASA Tech. Memo. 86369, Jan. 1985.

7. D. E. Eckhardt and L. D. Lee, "A theoretical basis for the analysis of multiversion software subject to coincident errors," *IEEE Trans. Software Eng.*, vol. SE-11, no. 12, pp. 1511-1517, 1985.

8. J. C. Knight and N. G. Leveson, "An experimental evaluation of the assumption of independence in multiversion programming," *IEEE Trans. Software Eng.*, vol. SE-12, pp. 96-109, Jan. 1986.

9. W. J. Krzanowski and D. Partridge, "Software diversity: practical statistics for its measurement and exploitation," Res. Rep. 324, Dept. Computer Science, University of Exeter (submitted to *Information & Software Technology*).

10. B. Littlewood and D. R. Miller, "Conceptual modelling of coincident failures in multiversion software," *IEEE Trans. Software Eng.*, vol. SE-15, no. 12, pp. 1596-1614, Dec. 1989.

11. D. Partridge, "Neural network differences quantified," *Neural Networks*, vol. 9, no. 2, pp. 263-271, 1996.

12. D. Partridge and W. B. Yates, "Engineering multiversion neural-net systems," *Neural Computation*, vol. 8, no. 4, pp. 869-893, 1996.

13. D. Partridge and W. B. Yates, "Data-defined problems and multiversion neural-net systems," *Journal of Intelligent Systems*, (in press).

Appendix A

Given that the a_i, the frequency distribution of failures in output category i, are arranged appropriately, and that $\frac{m+1}{n}, \ldots, \frac{m+1}{n}, \frac{m}{n}, \ldots, \frac{m}{n}$ are the distributions in the most diverse case, then d the distance of the actual distribution from the most diverse for a given input is:

$$d = \sum_{i=1}^{j} \sqrt{a_i \times \frac{m+1}{n}} + \sum_{j+1}^{c} \sqrt{a_i \times \frac{m}{n}}$$

We can now evaluate d for each row of the incidence matrix of Table 3 in which there is at least one failure, noting that $n \leq c = 5$ here.
In the first row we obtain $(n = 5)$

$$d = \sqrt{\frac{3}{5}\frac{1}{5}} + \sqrt{\frac{1}{5}\frac{1}{5}} + \sqrt{\frac{1}{5}\frac{1}{5}} = \frac{2+\sqrt{3}}{5} = 0.745$$

in the second row $(n = 3)$

$$d = \sqrt{\frac{1}{3}\frac{1}{3}} + \sqrt{\frac{2}{3}\frac{1}{3}} = 0.804$$

in the fourth row $(n = 5)$

$$d = \sqrt{\frac{2}{5}\frac{1}{5}} + \sqrt{\frac{1}{5}\frac{1}{5}} + \sqrt{\frac{2}{5}\frac{1}{5}} = 0.766$$

in the fifth row $(n = 1)$

$$d = \sqrt{\frac{1}{1}} = 1.0$$

in the sixth row $(n = 5)$

$$d = \sqrt{\frac{1}{5}1} = 0.447$$

and in the seventh row ($n = 4$)

$$d = \sqrt{\frac{1}{4}\frac{1}{1}} = 0.5$$

and in the ninth row ($n = 3$)

$$d = 3\sqrt{\frac{1}{3}\frac{1}{3}} = 1.0$$

With overall diversity, OD, defined as a *weighted* version of coincident-failure diversity, CFD:

$$OD = \sum_{n=1}^{N} \left(\frac{N-n}{N-1}\right) f_n w_n$$

where w_j is the average of the d values for all rows with j failures.

So here $w_1 = 1, w_2 = 0, w_3 = \frac{1}{2}(0.804 + 1.0) = 0.902, w_4 = 0.5$ and $w_5 = \frac{1}{3}(0.745 + 0.766 + 0.447) = 0.653$ and

$$OD = \frac{1}{6}\times 1 + \left(\frac{N-3}{N-1}\right)\times\frac{1}{6}\times 0.902 + \left(\frac{N-4}{N-1}\right)\times\frac{1}{6}\times 0.5 + \left(\frac{N-5}{N-1}\right)\times\frac{3}{6}\times 0.653.$$

When the value for N is substituted, we thus obtain

$$OD = \frac{1}{6} + \frac{2}{4}\times\frac{1}{6}\times 0.902 + \frac{1}{4}\times\frac{1}{6}\times 0.5 = 0.26$$

Appendix B

The following is a sample calculation of the g_5 value for the nine-version character recognition system under the assumption that the failure categories are equally likely.

$$\begin{aligned} c &= 25 \\ N &= 9 \\ k &= 4 \\ n &= k + 1 = 5 \end{aligned}$$

$$\begin{aligned} & \textit{equal distribution of failures} \\ \frac{1}{c^n} &= \frac{1}{25^5} \end{aligned}$$

$$
\begin{aligned}
\textit{set } \mathcal{D} \quad \textit{is} \quad & 1,1,1,1,1,0,...,0 \textit{ with } \begin{pmatrix} 25 \\ 5 \end{pmatrix} \textit{ distinct replicates,} \\
& 2,1,1,1,0,0,...,0 \textit{ with } \begin{pmatrix} 25 \\ 4 \end{pmatrix} \times 4 \textit{ distinct replicates,} \\
& 2,2,1,0,0,0,...,0 \textit{ with } \begin{pmatrix} 25 \\ 3 \end{pmatrix} \times 3 \textit{ distinct replicates,} \\
& 3,1,1,0,0,0,...,0 \textit{ with } \begin{pmatrix} 25 \\ 3 \end{pmatrix} \times 3 \textit{ distinct replicates,} \\
& 3,2,0,0,0,0,...,0 \textit{ with } \begin{pmatrix} 25 \\ 2 \end{pmatrix} \times 2 \textit{ distinct replicates,} \\
\textit{which} \quad \textit{gives} \quad & 5 \text{ values of } n_1!\ldots n_c! \\
\textit{of} \quad & 1,2,4,6,12, \textit{ respectively}
\end{aligned}
$$

$$
\begin{aligned}
\textit{hence} \quad & \quad g_5 \textit{ is} \\
\frac{5!}{25^5} \sum_{\mathcal{D}(n)} \frac{1}{n_1!\ldots n_c!} \quad = \quad & \frac{5!}{25^5} \times \\
& (\frac{1}{1}^{25} C_5 + \frac{1}{2}^{25} C_4 \times 4 + \frac{1}{4}^{25} C_3 \times 3 + \frac{1}{6}^{25} C_3 \times 3 + \frac{1}{12}^{25} C_2 \times 2) \\
= \quad & \frac{120}{9765625} \times 81355 \\
= \quad & 0.9997
\end{aligned}
$$

	number of versions, N						
	4	5	7	9	11	15	50
$E(\theta)^2$, $p(1)^2$	0.25	0.36	0.51	0.60	0.67	0.75	0.92
$E(\theta^2)$	0.25	0.36	0.51	0.60	0.67	0.75	0.92
$p(2)$	0.17	0.30	0.48	0.58	0.65	0.74	0.92
CFD	0.66	0.50	0.33	0.25	0.20	0.14	0.04
DFD	1.0	1.0	1.0	1.0	1.0	1.0	1.0
majority in agreement correct	100%	100%	100%	100%	100%	100%	100%
majority correct	0%	0%	0%	0%	0%	0%	0%

Table 4: Diversity estimates and system reliabilities

	$M = 10000$			$M = 9878$		
version	failures	successes	prob(succ)	failures	successes	prob(succ)
1	26	9974	99.74%	26	9852	99.74%
2	133	9867	98.67%	15	9863	99.85%
3	15	9985	99.85%	15	9863	99.85%
4	85	9915	99.15%	85	9793	99.14%
5	172	9828	98.28%	142	9736	98.56%
6	180	9820	98.20%	123	9755	98.75%

Table 5: test results on the six Modula2 versions

no. of versions	$M = 10000$		$M = 9878$	
failing, n	occurrences	p_n	occurrences	p_n
0	9609	0.960900	9606	0.972464
1	246	0.024600	196	0.019842
2	113	0.011300	61	0.006175
3	17	0.001700	0	0.000000
4	1	0.000100	1	0.000101
5	0	0.000000	0	0.000000
6	14	0.001400	14	0.001417

Table 6: the coincident-failures for the two tests of the six Modula2 versions

$M = 9878$						
version	failures	successes	prob(succ)	n	occurrences	p_n
1	18	9860	99.82%	0	9748	0.986839
2	56	9822	99.43%	1	107	0.010832
3	45	9833	99.54%	2	14	0.001417
4	10	9868	99.90%	3	1	0.000101
5	46	9832	99.53%	4	3	0.000304
				5	5	0.000506

Table 7: the test of the five Prolog versions

version set	$M = 9878$				
	CFD	average	maximum	minimum	p_{maj}
6 Modula2	0.901	99.31%	99.85%	98.56%	99.85%
5 Prolog	0.913	99.65%	99.90%	99.43%	99.91%

Table 8: diversity data from the two version sets

$M = 4000$								
	$pick_{CFD}$ system				$pick_{DFD}$ system			
n	g_n	t_n	occurrences	p_n	occurrences	p_n	t_n	g_n
0	—	—	1132	0.2830	420	0.1050	—	—
1	—	0.7625	1691	0.4228	672	0.1680	0.8386	—
2	—	0.1346	299	0.0747	771	0.1928	0.1098	—
3	—	0.0461	166	0.0415	932	0.2330	0.0305	—
4	—	0.0229	147	0.0367	468	0.1170	0.0125	—
5	0.7515	0.0174	**161**	**0.0403**	**184**	**0.0460**	0.0047	0.9185
6	0.4252	0.0061	**127**	**0.0318**	**141**	**0.0352**	0.0032	0.6312
7	0.0000	0.0059	**110**	**0.0275**	**147**	**0.0367**	0.0006	0.0544
8	—	0.0029	73	0.0182	150	0.0375	0.0000	—
9	—	0.0014	94	0.0235	115	0.0288	0.0001	—

Table 9: failure results for two nine-version systems

$M = 4000$								
system	CFD	DFD	OD	average	maximum	minimum	p_{maj}	p_{mia}
$pick_{CFD}$	0.814	0.942	0.776	79.81%	88.62%	33.77%	85.87%	90.25%
$pick_{DFD}$	0.709	0.969	0.673	69.92%	88.62%	33.77%	81.57%	88.22%

Table 10: the benefits of distinct-failure diversity in two nine-version systems