Predictive metric for likely feasibility of program paths

N Malevris, D F Yates and A Veevers*

The paper advances a claim about the nature of infeasible paths in programs and provides significant statistical evidence to support its validity. The characteristics of the claim permit the definition of an easily derived metric that, in view of the results of the statistical analysis presented, should provide a good predictor of the likely feasibility of program paths.

software testing, software metric, infeasible paths, program paths

The existence of infeasible program paths, paths whose execution cannot be caused by any data set, was documented at least as long ago as 1972[1]. Since then, there has been substantial recognition of the problems that these paths introduce[2-6]. The problems are perhaps most acute in connection with path-oriented testing techniques, that is, with testing techniques that involve:

- (1) selecting a set of paths through the program under test
- (2) deriving test data that will drive execution of the program down each selected path in turn

In the application of such a testing technique, one or more paths selected at step (1) can be infeasible. In this case, steps (1) and (2) must be performed iteratively, the extent of the iteration can be considerable, and the implications of this in respect of time, effort, and cost correspondingly substantial. Given this situation, together with the human characteristic to try to remove any obstacle that impedes attaining a desirable goal, surprisingly few authors have addressed the problems associated with infeasible paths[7,8]. Perhaps this state of affairs in some measure reflects the difficulties inherent in such problems, the extent of which is encapsulated in the result proved by Weyuker[9], which states that the problem of determining the feasibility of a program path is undecidable. Notwithstanding this, the fact that relatively little research aimed at combatting infeasible paths has been performed must be viewed with some consternation.

Two of the few papers that do directly address the issues associated with path feasibility are those of Woodward *et al.*[10] and Hedley and Hennell[8]. The first reports the authors' experience in using allegations to circumvent the generation of infeasible paths. Despite some success with the approach, it is concluded that 'it would undoubtedly be better to tackle the problem by other means', of which three are tentatively suggested, which rely respectively on improved language design, imposition of programming standards, and program optimization. The last two depend on distinguishing and categorizing infeasible paths and as such, Weyuker's result[9] is likely to preclude their efficacy. The first suggestion, as claimed by the authors themselves, 'may in the long term be the best answer', but at present, this is a moot point. The second paper[8] studies the occurrence of infeasible paths in a subset of the NAG (Numerical Algorithms Group) FORTRAN Library. Specifically, a largely successful attempt is described to categorize those situations that gave rise to the infeasible paths detected during the study. It is unfortunate, and once again a legacy of Weyuker's result, that feasible paths can also occur in the situations categorized.

A path generation strategy that attempts to minimize the time and cost overheads of branch testing was proposed[11]. The strategy is based on three observations: one about the nature of infeasible paths and the other two relating to factors that influence the time and cost overheads associated with branch testing. The first observation provides the motivation for this paper. It will be shown that an extended form of this observation leads to an assertion about the likely relative feasibility of program paths, that this assertion is statistically valid, and that its validity permits the definition of a metric for predicting the likely feasibility of program paths.

OBSERVATION, ASSERTION, AND METRIC

It was observed[11] that if Π is a path through a program, and Π involves $q>0$ predicates, then:

for Π to be feasible, all q predicates must be consis-

Department of Computer Science, University of Liverpool, Chadwick Building, PO Box 147, Liverpool L69 3BX, UK.
*Department of Statistics and Computational Mathematics, University of Liverpool, Victoria Building, PO Box 147, Liverpool L69 3BX, UK.
This paper has been held over from the January/February Special Issue on Software Quality Assurance.

 0950-5849/90/020115

tent, whereas infeasibility of Π requires as few as two predicates to be inconsistent.

Although correct, this observation is incomplete as it fails to take account of self-inconsistent predicates. Such predicates, of which two examples are:

$X>0.0$ **AND** $X<0.0$
$Y \geq W$ **OR** $W>Y$

are those whose logic precludes one or more possible outcomes of any test that involves them. Clearly, a path that involves a self-inconsistent predicate can be infeasible even when that predicate is the only one on the path, and, consequently, the above observation is extended as follows:

for a program path, Π, to be feasible, all q predicates must be both consistent and self-consistent, whereas infeasibility of Π requires either as few as two predicates to be inconsistent or one predicate to be self-inconsistent.

From this extended observation it is only a simple step to assert that:

if Π_1 and Π_2 are paths through a program P, involving $r>0$ and $s>r$ predicates, respectively, then Π_1 is more likely to be feasible than Π_2.

If this assertion can be substantiated, then the number of predicates involved in a program path can justifiably be used as an indicator, or metric, for the feasibility of that path.

STATISTICAL JUSTIFICATION OF ASSERTION

As a first step towards determining the validity of the assertion, it is appropriate to establish whether or not there is any association between the feasibility of a path and the number of predicates it involves. To investigate this, data was collected from a sample of 583 paths generated by the authors in performing branch testing and LCSAJ (Linear Code Sequence And Jump) testing[10] on a set of 36 subroutines chosen at random from the NAG Mark 11 FORTRAN Library. The number of paths in the sample involving q predicates that were determined to be feasible or infeasible for observed values of q is presented in Table 1.

Table 1. Feasible and infeasible paths in sample

No of predicates	Total	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Feasible	185	17	21	22	21	33	25	17	12	7	6	3	0	1	0	0	0
Infeasible	398	4	9	13	24	20	45	52	38	69	47	28	13	10	5	17	4
No of paths in sample	583	21	30	35	45	53	70	69	50	76	53	31	13	11	5	17	4

Table 2. Merged sample

No of predicates	Total	1	2	3	4	5	6	7	8	9	10	11	12 or more
Feasible	185	17	21	22	21	33	25	17	12	7	6	3	1
Infeasible	398	4	9	13	24	20	45	52	38	69	47	28	49
No of paths in sample	583	21	30	35	45	53	70	69	50	76	53	31	50

If, in fact, there is no association between the feasibility of a path and the number of predicates it involves, then it would be expected that approximately equal proportions of feasible paths will occur for all values of $q \geq 1$. To test formally the hypothesis, H_0, of no association, that is, of there being equal proportions of feasible paths for all $q \geq 1$, a χ^2 test is performed on the entries, derived from Table 1 by merging its last five columns, given in Table 2. (Any standard statistical text[12] shows the necessity of this procedure.)

Standard statistical texts show that for each integral $\nu>0$, referred to as 'the number of degrees of freedom', there exists a value $\chi^2_\nu(\alpha)$ such that if χ^2 (calculated from the sample[12]) $\geq \chi^2_\nu(\alpha)$, then the hypothesis H_0 of no association, that is independence of the two categorizing variables, is rejected with significance probability α, and is accepted otherwise. In essence, χ^2 is a measure of the departure of the sample from what would be expected if H_0 were true, and it is the calculation of χ^2 and its subsequent comparison with $\chi^2_\nu(\alpha)$ that constitutes the χ^2 test.

In the present case, $\nu = 11$, and the categorizing variables are the feasibility of a path and the number of predicates it involves. Using the entries in Table 2, a value of $\chi^2 = 145.77$ is obtained, and comparing this value with standard χ^2 tables for $\nu = 11$, $\chi^2_{11}(0.005) = 26.755$. That is, the result $\chi^2 = 145.77$ is significant at the 0.5% level, and so H_0 is confidently rejected. Thus it is concluded that the proportion of feasible paths among those that possess q predicates is not the same for each value of q, and therefore there must be some form of dependency between a path's feasibility and the number of predicates it involves.

The assertion under investigation states that, in the long run, the proportion p_s of feasible paths with s predicates is less than the corresponding proportion p_r of paths that possess $r<s$ predicates, for all valid values of r and s. Each long run proportion, p_q, which is the probability that a path with q predicates is feasible, can be estimated from the sample by the corresponding observed proportion f_q. The plot of f_q against q, given in Figure 1, indicates a general decrease in f_q as q increases, and thereby provides support for the assertion (the line $f_q = 0.32$ indicates the mean value of f_q for the sample).

Parametric functions that might be appropriate for modelling this situation are polynomials in q and the exponentially decaying function

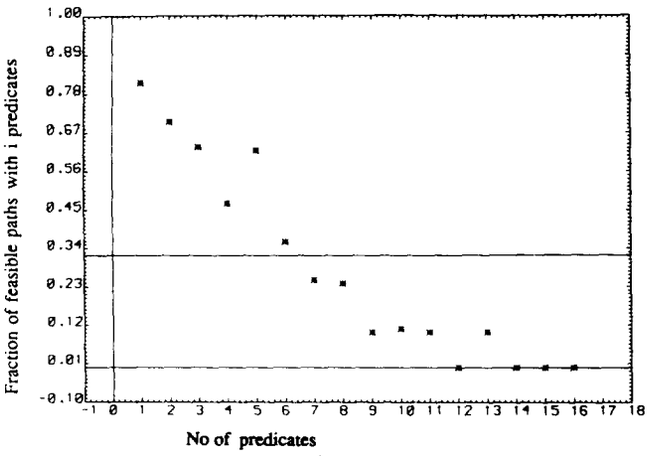

Figure 1. Plot of proportions of feasible paths versus no of predicates

$$p_{q+1}=Cp_q=e^{-\beta}p_q \quad q=1,2, \ldots \tag{1}$$

where the constant C is to be determined along with p_1. Although a polynomial in q might provide a good model for f_q in the range $1 \leqslant q \leqslant 12$, when q is continuously incremented, f_q either increases or decreases monotonically without bound. A polynomial model is thus rejected.

To investigate the appropriateness of the exponetially decaying function, a least squares fit of equation (1) to the observed values of f_q was performed. The function obtained by the fitting process was

$$p_{q+1}=e^{-0.1988}p_q \text{ with } p_1=1.0703e^{-0.1988} \tag{2}$$

This corresponds to the continuous function

$$p_q=1.0703e^{-0.1988q}$$

which, together with the values of f_q, $q = 1, 2, \ldots, 16$, is plotted in Figure 2.

The fitting process revealed a value of $Z = 0.088$ for the sum of the squares of the residuals at q, $q = 1, 2, \ldots, 16$. This is significantly less than the corresponding value of 1.209 obtained from the model predicting a

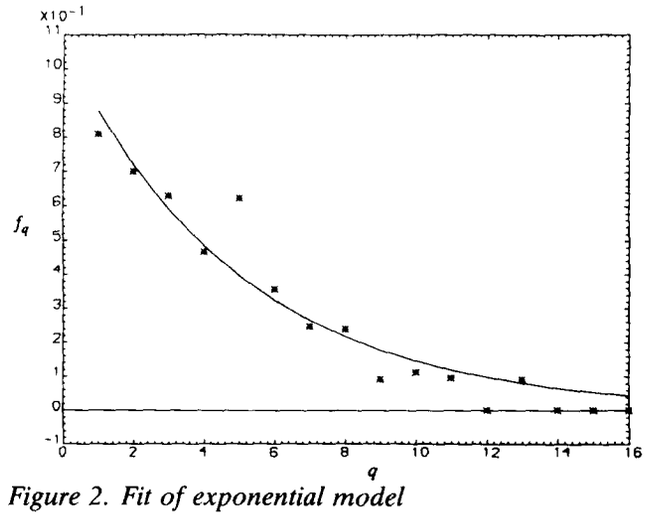

Figure 2. Fit of exponential model

constant proportion of feasible paths for all q, thereby indicating that equation (1) provides a more suitable model for explaining the observed values of f_q. To determine whether or not equation (2) is optimal in its class (that of discrete exponentially decaying functions with polynomial exponents), a discrete exponentially decaying function with a slightly more complex exponent was investigated. Correspondingly, a least squares fit of the parametric function

$$p_{q+1}=p_qe^{-\mu q-\lambda} \quad q=1, 2, \ldots \tag{3}$$

to the observed values of f_q was performed. Here μ and λ are constants to be determined together with p_1.

Although the least squares fit of equation (3) necessarily realised a smaller value for the sum of the squares of the residuals, the application of an F-test[12] showed that the reduction achieved was not significant. The implication here is that increasing the complexity of the exponent over and above that in equation (2) does not significantly improve the fit obtained. In view of this, the (small) size of Z, and that a value of p_0 (1.0703) close to unity is predicted for a path with no predicates (this would be expected of any good model), it is concluded that equation (2) provides an appropriate model for explaining the nature of the relationship between the observed values of f_q. It is therefore claimed that the above provides strong supporting evidence that path feasibility does indeed tend to decay with an increasing number of predicates.

SUMMARY AND CONCLUSION

A criterion, or metric, for gauging the likely feasibility of program paths has been proposed. The effectiveness of this metric depends on the validity of the assertion that program paths that possess fewer predicates are more likely to be feasible than those that possess more.

As a result of a χ^2 test performed on a sample of 583 program paths, it was concluded with extreme confidence that there is some form of dependency between path feasibility and predicate involvement. To investigate the possible relationship, a least squares fit of the parametric function $f_q = Ke^{-\lambda q}$ to the sample data points was performed. The function $f_q = 1.0703e^{-0.1988q}$ obtained resulted in a value of Z (sum of the squares of the residuals at the data points) that was sufficiently small, compared with that for others in the general exponential class of models, for it to be concluded that this function represented an acceptable model for explaining the relationship observed between the sample data points.

One consequence of the result that the problem of determining the feasibility of a program path is undecidable[9] is the impossibility of devising a criterion that will guarantee to distinguish between feasible and infeasible paths. Therefore, any metric that is capable of distinguishing between certain feasible and infeasible paths must be adjudged to be heuristic. The authors contend that the results presented provide considerable evidence to support the assertion made in the second

section about the nature of infeasible paths and further contend that these same results are equally supportive of the metric

Q = the number of predicates involved in a path

being a good heuristic for assessing a path's feasibility.

ACKNOWLEDGEMENT

The authors thank NAG Ltd for permission to access the source code version of their Mark 11 FORTRAN Library.

REFERENCES

1 **Brown, J R** 'Practical application of automated software tools' *TRW Report TRW-SS-72-05* TRW Systems, Redondo Beach, CA, USA (1972)

2 **Huang, J C** 'An approach to program testing' *ACM Comput. Surv.* Vol 7 No 3 (1975) pp 113–128

3 **Howden, W E** 'The theory and practice of functional testing' *IEEE Software* Vol 2 No 5 (1985) pp 6–17

4 **White, L J** 'Software testing and verification' in **Yovits, M C (ed)** *Advances in computers* Academic Press, Orlando, FL, USA Vol 26 (1987) pp 335–391

5 **Adrion, W R, Branstad, M A and Cherniavski, J C** 'Validation, verification, and testing of computer software' *ACM Comput. Surv.* Vol 14 No 2 (1982) pp 159–192

6 **Prather, R E** 'Theory of program testing - an overview' *Bell Syst. Tech. J.* Vol 62 No 10(2) (1983) pp 3073–3105

7 **Chellappa, M** 'Nontraversible paths in a program' *IEEE Trans. Soft. Eng.* Vol 13 No 6 (1987) pp 751–756

8 **Hedley, D and Hennell, M A** 'The causes and effects of infeasible paths in computer programs' in *Proc. 8th ICSE* London, UK (1985) pp 259–266

9 **Weyuker, E J** 'The applicability of program schema results to programs' *Int. J. Comput. Inf. Sci.* Vol 8 (1979) pp 387–403

10 **Woodward, M R, Hedley, D and Hennell, M A** 'Observations and experience of path analysis and testing of programs' in *Proc. IEEE Workshop Software Testing and Test Documentation* Fort Lauderdale, FL, USA (1978) pp 70–96

11 **Yates, D F and Hennell, M A** 'An approach to branch testing' in *Proc. 11th Int. Workshop Graph Theoretic Techniques in Computer Science* Wurtzburg, FRG (1985) pp 421–433

12 **Chatfield, C** *Statistics for technology* Chapman and Hall, New York, NY, USA (1981)