An Analysis of Output-Based Partition Testing for Heuristic Classification Expert Systems

I. A. Zualkernan
Yuan-Jing Lin
Dept. of Electrical and Computer Engineering
Pennsylvania State Univ.
State College, PA16801
zualkern@pacific.cmpe.psu.edu

Abstract

Although a host of testing methods have been proposed for expert systems, little work has been done to compare effectiveness of these methods. This paper presents an analysis of various parameters that govern the effectiveness of output based partition testing strategies for heuristic classification expert systems.

Introduction

Expert systems are programs that mimic some aspects of human expertise to solve ill-structured problems. In this paper we explore the effectiveness of partition testing methods for a class of expert systems based on *heuristic classification* (Clancey 1985).

The motivation for studying partition testing for expert systems is based on the following observations.

- Although a number of testing methods (Myers 1979) (Suwa, Scott, & Shortliffe 1982) (Nguyen et al. 1987) (Ould & Unwin 1987) (Tsai & Zualkernan 1990) (Gupta 1991) (Tsai, Zualkernan, & Kirani 1992a) have been proposed for testing expert systems, recent studies (Zualkernan, Tsai, & Kirani 1992) (Tsai, Kirani, & Zualkernan 1992b) (Tsai, Kirani, & Zualkernan 1993) suggest that conventional testing methods such as random and partition testing are much more effective at catching failures than consistency and completeness checking methods.
- Although effective, random testing requires a large number of test cases (Beizer 1990). Since the generation and evaluation of test cases for expert system is very expensive (Zualkernan, Tsai, & Kirani 1992), we need to look for better partition strategies.
- Testing plays a crucial role in the life-cycle of expert systems which is exploratory (Hayes-Roth et al. 1983) or spiral (Boehm 1987) (Zualkernan 1991) in nature. We need to consider partition testing as a vehicle for both *confidence establishing* and *failure exposing* capabilities.
- To be effective, partition testing methods have to perform *considerably* better than random testing because every time specification, design or code changes due to spiral nature of the development process, the partition may be recalculated thus incurring considerable additional cost. Alternatively, partition testing methods could be made less sensitive to such changes.

Previous Work

Partition testing has been compared with random testing for conventional software (Duran & Ntafos 1984) (Hamlet & Taylor 1990) (Loo, Tsai, & Tsai 1989) (Weyuker & Jeng 1991). Much of this work is relevant to testing of expert systems. Duran and Ntafos (Duran & Ntafos 1984) found the counter intuitive result that random testing performed almost as well as partition testing. This result was confirmed by further experimentation by (Hamlet & Taylor 1990). These studies show that partition testing can be better or worst than random testing depending on how the inputs producing failures[1] are concentrated within partitions. This implies that partition testing may be most effective when partition definitions are failure-based (Hamlet & Taylor 1990).

In applying these previous results to partition testing for expert systems, we have to look at their assumptions. Three types of assumptions about the partitions are relevant: overlap, size and defect distribution. Most previous work assumes that the partitions do not overlap. Some previous work (Duran & Ntafos 1984) (Hamlet & Taylor 1990) (Weyuker & Jeng 1991) assumes that the partition sizes are the same. Others (Loo, Tsai, & Tsai 1989) assume that partitions sizes can be different. Most prior work assumes that the partitions are *homogeneous.* A partition is homogeneous if most elements either fail or succeed.

Most previous work have used what we term as the conservative partitioning strategy (CPS) in which only

[1] In this paper we adopt the IEEE standard terminology of *error, fault* and *failure.* Errors represent human mistakes which can result in faults in a system. A fault may cause a system to fail on multiple inputs, but each failure can potentially lead to the discovery of a new fault.

one element is picked from each partition. CPS strategy is reasonable if one assumes *homogeneous* partitions. For non-homogeneous partitions, some (Loo, Tsai, & Tsai 1989) have proposed to use a constant number for sampling.

The most common comparison criteria used for comparing testing strategies is the probability of catching at least one failure (Hamlet & Taylor 1990) (Weyuker & Jeng 1991). A testing strategy with a higher probability of detecting a failure is preferred. Others have argued, that while probability of detecting at least one failure may be a good measure for *confidence establishing*, a better measure for *failure detection* may be the expected number of failures caught (Duran & Ntafos 1984) (Loo, Tsai, & Tsai 1989) and the expected number of faulty partitions (Loo, Tsai, & Tsai 1989).

Heuristic Classification Expert Systems

Clancey (clancey 1985) analyzed a number of expert systems and found that they all passed through the recognized phases of data abstraction, heuristic mapping into hierarchy of pre-enumerated solutions and refinement within this hierarchy. He called this type of reasoning heuristic classification and showed that earlier expert systems such as MYCIN seem to use heuristic classification.

For this paper, the important property of expert systems using heuristic classification (referred to as HCE systems from here on) is the existence of a pre-enumerated hierarchy of solution classes.

Partition Testing For HCE Systems

For partition testing, we make the following observations about the nature of partitions induced on the output space of an HCE system.

- Given a class hierarchy, we can choose to use partitions at any level of the classification tree.
- Partitions based on the class hierarchy are naturally induced by the development process. One does not need to recompute the partitions every time just for doing partition testing.
- There is no *a priori* reason to believe that the partitions induced by the class hierarchy are of the same size.
- There is no *a priori* reason to believe that the faults are uniformly distributed in a partition.
- There is no *a priori* reason to believe that the partitions are *homogeneous.*
- There is no *a priori* reason to believe that there should be a correlation between the failure rate and the size of a partition.

Most previous work has shown that under a variety of conditions the CPS strategy does not do significantly better than random testing (Duran & Ntafos 1984) (Hamlet & Taylor 1990). Hence, some have suggested that we exploit information about the nature of the failures in each partition in order to construct better strategies to detect *actual failures.* This approach seems feasible for constructing partition strategies for HCE. Since we are using the class hierarchy as the natural partitioning of the output space, different boundaries of partitions will exist depending on the level of the classification tree.

Formalization

In this section we present the notation to express our ideas more formally.

n The total number of the test cases in a test suite.

k The total number of partitions.

m_i The size of a partition i.

n_i The total number of the test elements sampled from a partition i. $\sum_{i=1}^{k} n_i = n$.

p_i The probability that an element is picked from partition i. $p_i = \frac{m_i}{\sum_{i=1}^{k} m_i}$

θ The failure rate for the whole domain.

θ_i The failure rate for partition i. $\theta = \sum_{i=1}^{k} p_i\theta_i$.

Based on the notation, we can describe various classes of partition strategies as follows:

exhaustive In exhaustive strategy each element in the domain is sampled. $k = \sum_{i=1}^{k} m_i$ and $(n_i = m_i = 1)$.

random In random testing sampling is proportional to the size of each partition $(n_i \propto m_i)$.

conservative partition (CPS) In conservative partition strategy only some constant elements are sampled from each partition ($n_i = c$ where c is a constant).

failure based partition (FPS) In this optimistic partition strategy some estimate of failure rate is used and the sampling in each partition is proportional to the number of failures in it ($n_i = cm_i\theta_i$ where c is a constant).

The criteria of determine 'goodness' of a testing strategy are given as follows (Loo, Tsai, & Tsai 1989):

P_p The probability of finding at least one failure for partition testing, $P_p = 1 - \prod_{i=1}^{k}(1-\theta_i)^{n_i}$.

P_r The probability of finding at least one failure for random testing, $P_r = 1 - (1-\theta)^n$.

EF_p The expected number of failures when using partition testing, $EF_p = \sum_{i=1}^{k} n_i\theta_i$.

EF_r The expected number of failures when using random testing, $EF_r = n\theta = n\sum_{i=1}^{k} p_i\theta_i$.

ED_p The expected number of faulty partitions for partition testing, $ED_p = \sum_{i=1}^{k}(1-(1-\theta_i)^{n_i})$.

ED_r The expected number of faulty partitions for random testing, $ED_r = \sum_{i=1}^{k}(1-(1-\theta_i)^{np_i})$.

Duran (Duran & Ntafos 1984), Hamlet (Hamlet & Taylor 1990), and Weyuker (Weyuker & Jeng 1991) use P_r and P_p to compare partition testing and random testing methods. Loo (Loo, Tsai, & Tsai 1989) use EF_p, EF_r, ED_p, ED_r. In order to determine the effectiveness of a partition strategy for HCE expert systems, we will use three composite metrics: $\alpha = \frac{EF_p}{EF_r}$, $\beta = \frac{ED_p}{ED_r}$, $\gamma = \frac{P_p}{P_r}$. Each metric compares a partition strategy against a random testing strategy. A good partition strategy maximizes α, β and γ.

Experiment

The performance of FPS and CPS strategies depends on failure rate and partition size distributions.

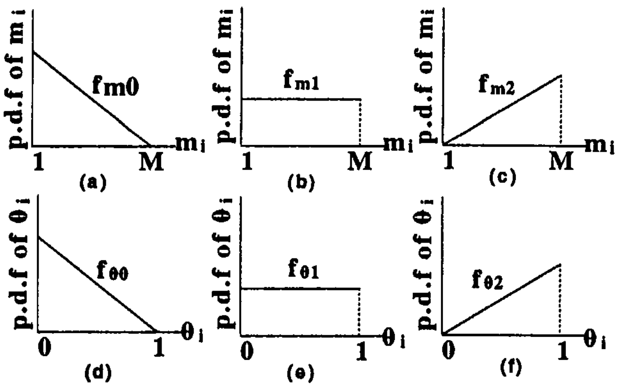

Figure 1:Probability distribution functions for partition size (mi) and failure rate (θi)

Qualitatively, we can think of three types of failure rate distributions: mostly faulty partitions, uniformly faulty partitions and mostly non-faulty partitions. When the development starts, we may expect most of the partitions to have high failure rates. Gradually, the rate may become uniform across partitions. Before we deliver the system, we may expect most partitions to be fault free. Similarly, partition size distribution can be divided into categories of mostly large, uniform sized, and mostly small partitions.

To simulate various situations related to failure rate distribution, we used the three probability density functions (P.d.f) shown in Figure 1 (d), (e), and (f). We picked the simplest functions, $f_{\theta 0}$, $f_{\theta 1}$, and $f_{\theta 2}$, to represent the conditions of mostly faulty, uniformly faulty, and mostly non-faulty partitions. In Figure 1 (a), (b), and (c), f_{m0}, f_{m1}, and f_{m2} were chosen to describe situations with mostly large-size, uniform-size, and mostly small-size partitions.

The statistical properties of these probability distribution functions are described in Table 1, where m denote the number of elements in a partition and θ is the failure rate of a partition and the maximum partition size is M.

	P.d.f.	Mean	Variance
m	$f_{m0}(m) = \frac{2(M-m)}{(M-1)^2}$	$E_{m0} = \frac{M+2}{3}$	$\sigma^2_{m0} = \frac{(M-1)^2}{18}$
	$f_{m1}(m) = \frac{1}{M-1}$	$E_{m1} = \frac{M+1}{2}$	$\sigma^2_{m1} = \frac{(M-1)^2}{12}$
	$f_{m2}(m) = \frac{2(m-1)}{(M-1)^2}$	$E_{m2} = \frac{2M+1}{3}$	$\sigma^2_{m2} = \frac{(M-1)^2}{18}$
θ	$f_{\theta 0}(\theta) = 2(1-\theta)$	$E_{\theta 0} = \frac{1}{3}$	$\sigma^2_{\theta 0} = \frac{1}{18}$
	$f_{\theta 1}(\theta) = 1$	$E_{\theta 1} = \frac{1}{2}$	$\sigma^2_{\theta 1} = \frac{1}{12}$
	$f_{\theta 2}(\theta) = 2\theta$	$E_{\theta 2} = \frac{2}{3}$	$\sigma^2_{\theta 2} = \frac{1}{18}$

Table 1: Properties of P. d. f's for θ_i and m_i

Method

For each partition strategy we carried out monte-carlo simulations (Payne 1982) for the total 9 combinations of failure rate and partition size distributions. Each situation is represented by s_{ij} where i denotes the P.d.f used for partition size and j the P.d.f used for failure rate; e.g., s_{10} represents a situation where f_{m1} and $f_{\theta 0}$ are used.

For CPS, in each situation, we varied k from 1 to 100 and c from 1 to 50 for 11 different values: 1, 5, 10, 15, 20, 25, 30, 35, 40, 45 and 50. For each combination of k and c, 1000 experiments were conducted. In each experiment, we randomly generated an m_i based on f_{mi} and a θ_j based on $f_{\theta j}$ and calculated the values of α, β and γ. For each k, we calculated the means and variances for the three parameters. We arbitrarily assumed M (see Figure 1) to be 200. We carried out $9 \times 100 \times 11 \times 1000 = 9.9 \times 10^6$ experiments.

For FPS, the same procedure was repeated, except that c was varied from 1 to 10. We conducted $9 \times 100 \times 10 \times 1000 = 9 \times 10^6$ experiments for FPS.

Results

CPS Strategy Figures 2, 3 and 4 shows how α, β and γ vary with respect to k and c. We make the following observations.

1. $\alpha \approx 1$ for any number of partitions (see Figure 2).
2. Although $\beta \approx 1$ (see Figure 3) for the case with mostly small partitions (s_{0j}), CPS does offer some advantage in exposing partitions with failures. β seems to be better for uniform partitions (s_{1j}) than for mostly large partitions (s_{2j}). Within the uniform sized partitions, the most advantage is gained in partitions with mostly high failure rate (s_{12}), followed by uniform (s_{11}) and mostly low failure rates (s_{10}). The same is true for various types of partitions within mostly large partitions. The advantage gained by CPS over random testing varies from about 2% to 11%.
3. γ is maximized for low values of k and c and the advantage for CPS disappears for about $k = 23$ or

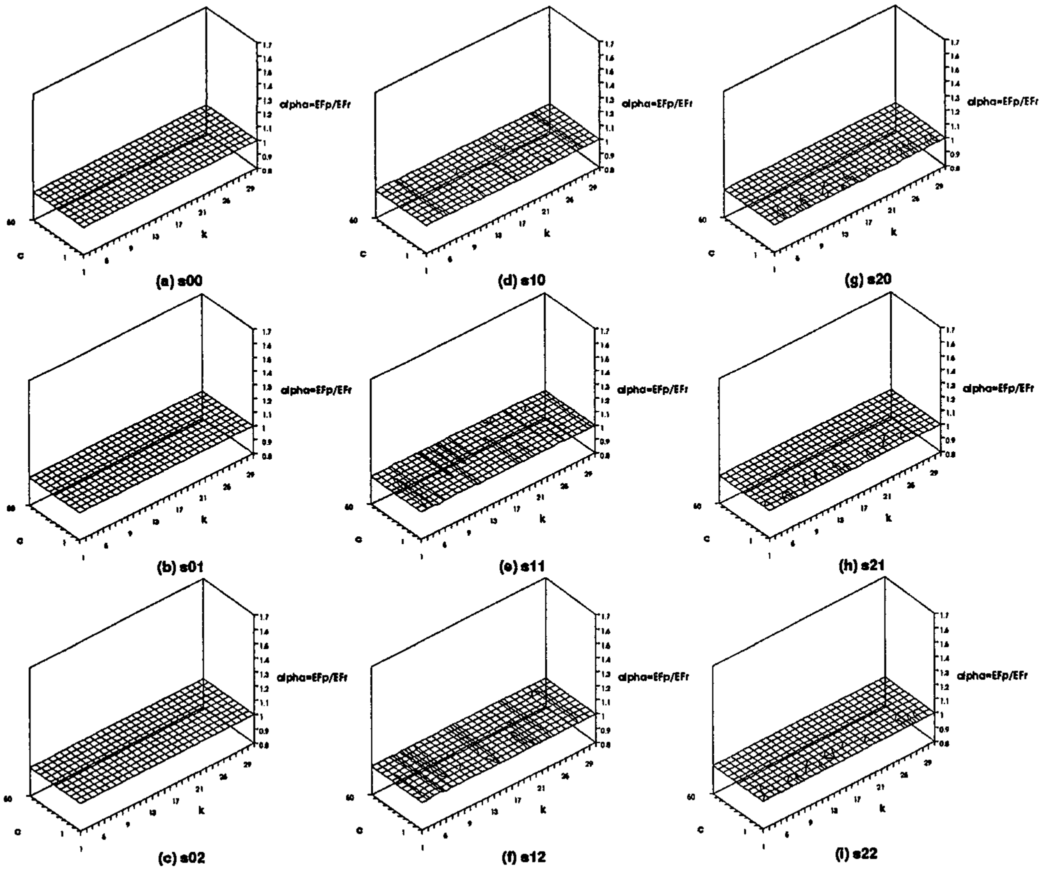

Figure 2: Relationship between k and c and α for Conservative Partition Strategy under various conditions (s_{ij}) of partition size distribution (f_m) and failure rate distribution (f_α).

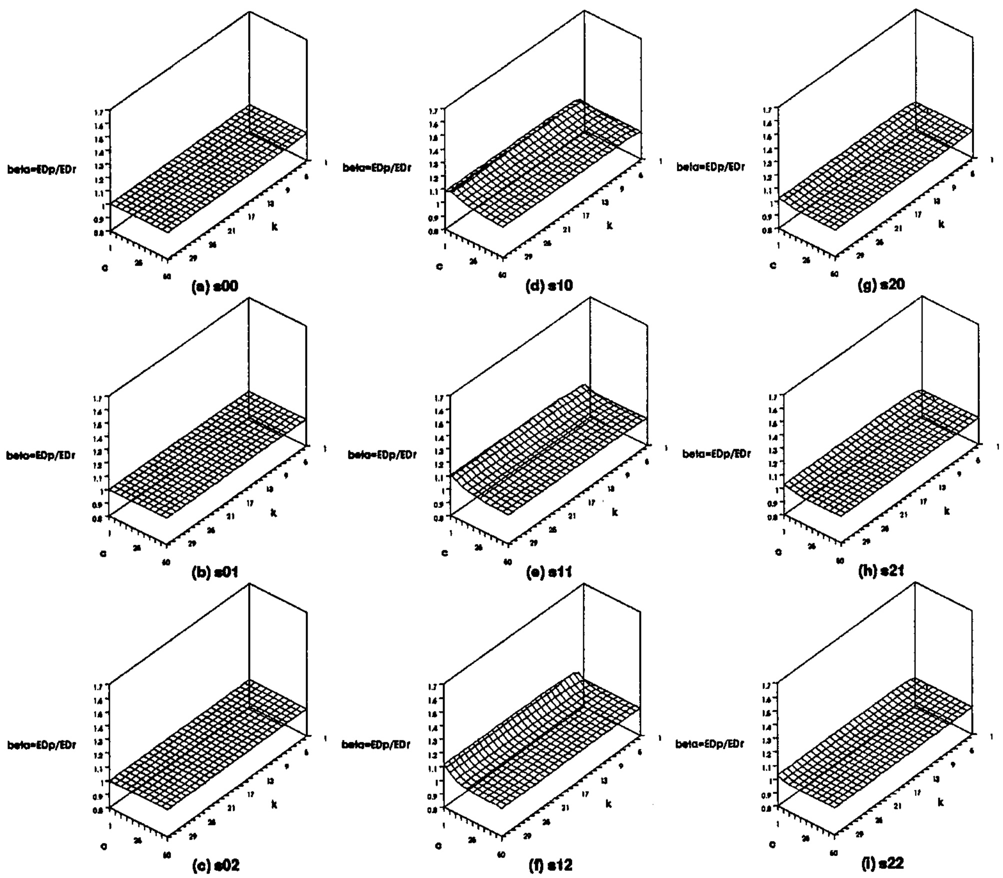

Figure 3: Relationship between k and c and β for Conservative Partition Strategy under various conditions (s_{ij}) of partition size distribution (f_m) and failure rate distribution (f_α).

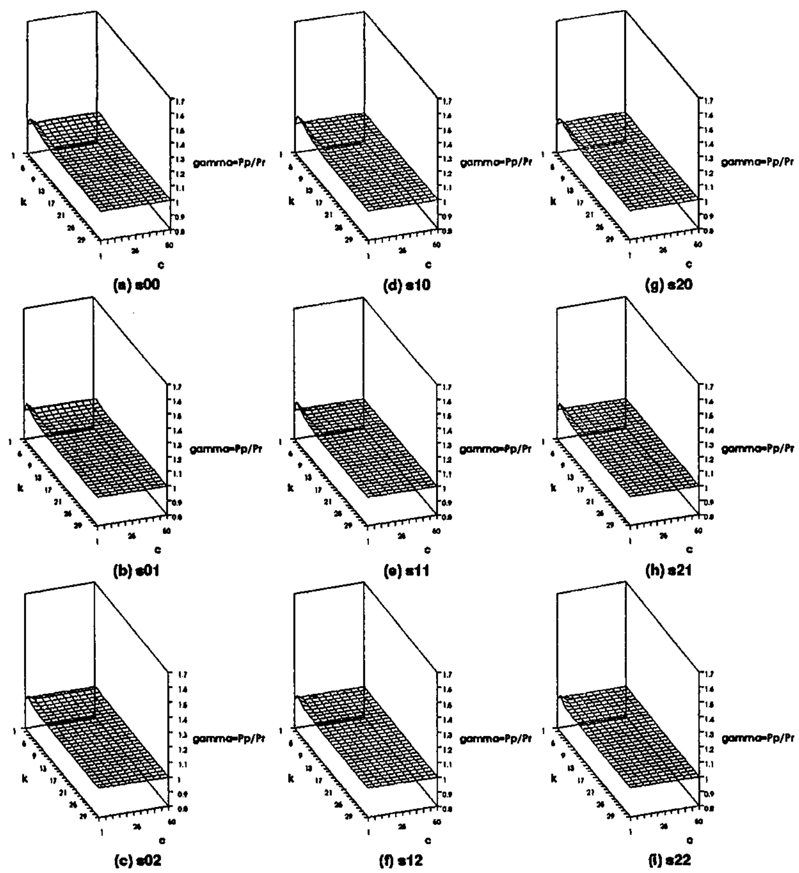

Figure 4: Relationship between k and c and γ for Conservative Partition Strategy under various conditions (s_{ij}) of partition size distribution (f_{mi}) and failure rate distribution ($f_{\lambda j}$).

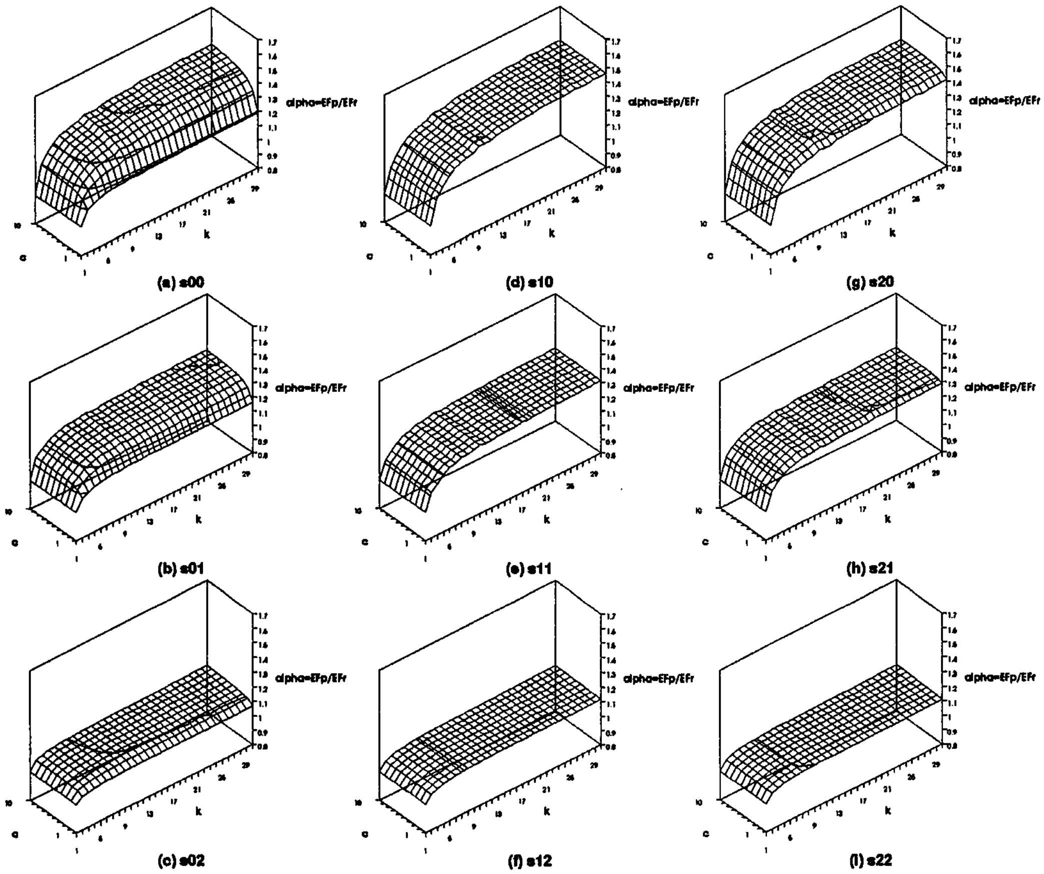

Figure 5: Relationship between k and c and α for Failure based Partition Strategy under various conditions (s_{ij}) of partition size distribution (f_{mi}) and failure rate distribution ($f_{\lambda j}$).

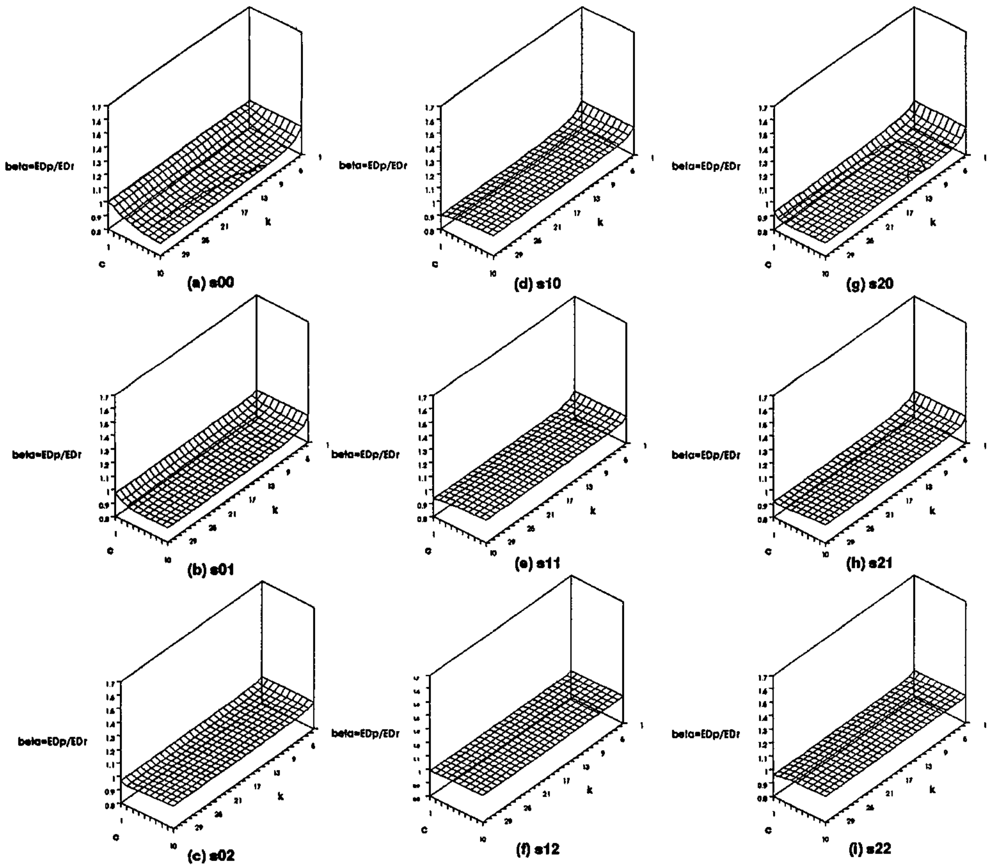

Figure 6: Relationship between k and c and β for Failure based Partition Strategy under various conditions (s_{ij}) of partition size distribution (f_{mi}) and failure rate distribution (f_{dj}).

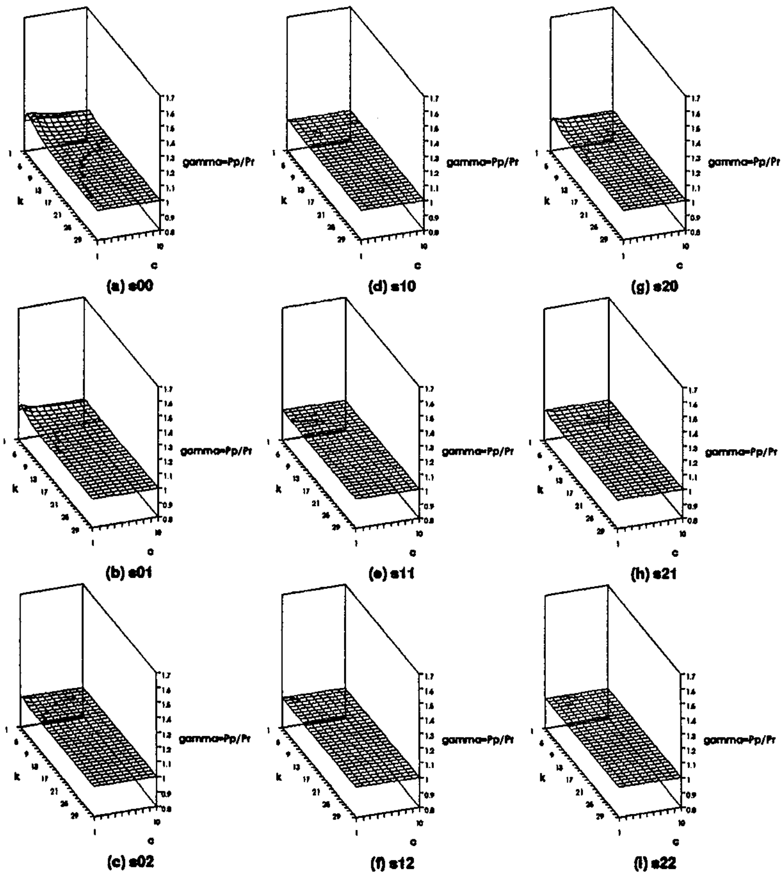

Figure 7: Relationship between k and c and γ for Failure based Partition Strategy under various conditions (s_{ij}) of partition size distribution (f_{mi}) and failure rate distribution (f_{dj}).

$c \geq 5$ (see Figure 4). The maximum advantage varies from about 4% to 7.4%. The behavior of γ also seems to be grouped according to the failure distribution. For high failure situations (s_{i2}) the advantage is the lowest (about 4%) and this advantage disappears most quickly (at $k \approx 10$) as k increases. γ for CPS seems to favor situations where the partitions are mostly non-faulty (s_{i0}).

FPS Strategies Figure 5, 6, and 7 show the behavior of α, β, and γ with respect to k and c. We make the following obervations:

- Overall Behavior:

1. As k and c increase, α quickly rises and then becomes almost asymptotic with respect to both c and k (see Figure 5). In all situations, $\alpha \geq 1$. α seems to be the highest for situations with mostly non-faulty partitions and the lowest for mostly faulty partitions.
2. Under most values of c, β is less than 1 (see Figure 6).
3. γ has a low maximum value as $c = 1$ across situations and quickly approaches 1 as k and c increase (see Figure 7).

- Results for a Fixed c ($c = 2$):

1. The FPS strategy offer an advantage from 8% to 45% over random testing in detecting more failures.
2. The advantage for FPS strategy grows initially and becomes asymptotic as k increases.
3. The situations with mostly non-faulty partitions, seem to mostly offer the highest advantage for α. The least advantage is gained in situations with high failure rates.
4. β becomes asymptotic for increasing k but is always less than 1 for all situations.

- Results With Respect to a Variable c:

1. Situations with mostly non-faulty partitions within a particular size distribution have the highest α across c and those with mostly faulty partitions have the lowest α across c.
2. Within a particular failure distribution, uniform sized partitions seem to have the highest but constant α.
3. Within a particular failure distribution, most advantage is gained by initially increasing c for partitions that have mostly small sized distribution.
4. As c becomes large, the differences between α due to partition size distribution tend to disappear.
5. With increasing c, β either increases or becomes constant in all situations except for s_{00}.
6. Situations with mostly non-faulty partitions tend to retain their advantage over the other situations. across c for the same type of failure distribution.
7. From mostly non-faulty partitions to uniformly faulty partitions, to mostly faulty partitions, the maximum γ gets *smaller* with respect to c.
8. As c increases, the advantage of FPS to detect at least one failure disappears. Further, the advantage disappears more quickly for mostly faulty partition.

Summary We summarize the results as follows:

- FPS strategies detect more failures than random testing under a variety of situations. However, if we increase the number of partitions (k) or the sampling constant (c), the marginal gain becomes very small.
- FPS strategies detect less faulty partitions than random testing under a variety of situations for reasonable values of c and k.
- FPS strategies have little advantage over random testing for detecting at least one failure. This advantage quickly disappears as c and k are increased.

Discussion and Implications

We carried out the analysis for conservative and failure based partition strategies. Conservative partition strategies do not perform better than random testing in detecting more failures but do perform slightly better in uncovering faulty partitions. Failure-based partition strategies tend to do better (up to 47%) than random testing in detecting failures but do worse (by 12%) than random testing in detecting faulty partitions. For acceptance testing, both partition strategies tend to do slightly better than random testing.

While CPS and FPS strategies have been used by expert system developers, an understanding of their limitations and tradeoffs is essential. For example, consider the test case selection strategy for evaluation of MYCIN (Yu et al. 1984). This strategy is an example of a slight categories of CPS; at least one test case was selected from each of the four variations of meningitis.

From our experiments, for $k = 4$, (see Figure 4) if only 4 cases were picked, the use of this 'diverse origins' strategy would have only increased γ by from 2% to 7%. Hence we are able to assign quantitative measures to qualitative notions such as the use of 'diverse origins.' We can also say that the use of this strategy would not have exposed any more failures than random testing. Similarly, under the best conditions, it would have exposed a maximum of 1.1×4 failures. If an estimate of number of expected failures had been available, a use of FPS strategy (with $c = 2$)in the same case would have resulted in $\alpha \approx 1.2$. In this situation no significant advantage would have been obtained because FPS would be expected to detect at best $0.2 \times 8 = 1.6 \approx 2$ more failures than random testing.

The above analysis based on the expriments conducted in this paper shows that the use of the 'diverse origins' strategy used to test MYCIN'S was really not better than random testing. Our experiments suggest that the use of such a strategy could have been beneficial if a failure based partition strategy were used but only for a larger number of partitions.

We have shown how our experiments can be used to make informed decisions about the use of partition testing strategies. Overall, the results are sobering. The partition testing strategies, even if based on perfect information about failure distribution do not perform miraclously better than random testing. The largest gain for detecting number of failures is about 47%. Surprisingly, the use of failure based partition testing in most reasonable circumstances *reduces* the expected number of partitions caught when compared with random testing.

More than anything, these experiments point out the importance of basing validation and verification of expert systems on a sounder and quantitative basis because *intuition* can often be misleading.

Conclusions

In this paper, we have presented a formal and empirical analysis of output based partition strategies for *heuristic classification* expert system. A preliminary analysis based on ideal distributions of failure rate and size distribution indicates statistically significant differences in the output metrics such as number of failures exposed, number of faulty partitions uncovered and the probability of detecting at least one failure.

If we can derive reasonable estimates of the relative sizes and the failure rates for a particular classification tree, it is possible to derive an optimal k or depth of the tree that maximizes α, β and γ. Since it is possible to derive multiple classification hierarchies for the same problem, information about maximizing α, β and γ can also be used to design the output classification tree for *testability*.

A quantitative analysis of the expert system testing methods will not only help us understand their real limitations but will also help us design better testing methods in the future.

References

Beizer, B. 1990. *Software Testing Techniques.* Van Nostrand Reinhold, 2nd edition.

Boehm, B. W. 1987. Industrial software metrics - top ten list. *IEEE Software*, 3 (9):84–85.

Clancey, W. J. 1985. Heuristic classification. *Artificial Intelligence*, 27.

Duran, J. W. and Ntafos, S.C. 1984. An evaluation of random testing. *IEEE Trans. Software Eng.*, 10 (4):438–444.

Gupta, U. G. 1991. *Validating and Verifying Knowledge-based Systems.* IEEE Computer Society Press.

Hamlet, D. and Taylor, R. 1990. Partition testing does not inspire confidence. *IEEE Trans. Software Eng.*, 16 (12):1402–1411.

Hayes-Roth, R. et al. 1983. *Building Expert Systems.* Addison-Wesley.

Myers, G. 1979. *The Art of Software Testing.* Prentice Hall.

Nguyen, T. A. et al. 1987. Knowledge base verification. *AI Magazine*, 65–79.

Ould, M. A. and Unwin, C. 1987. *Testing in Software Development.* Cambridge Univ. Press.

Payne, J. A. 1982. *Introduction to Simulation: Programming Techniques and Methods of Analysis.* McGraw-Hill Book Company.

Loo, P. S., Tsai, W. T., and Tsai, W. K. 1989. Random testing revisited. *Information and Software Technology*, 30 (7):402–417.

Suwa, M., Scott, A. C., and Shortliffe, E. H. 1982. An approach to verifying completeness and consistency in a rule-based expert system. *AI Magazine*, 16–21.

Tsai, W. T., Kirani, S., and Zualkernan, I. A. 1993. Evaluation of expert system testing methods. *Communications of the ACM - To appear.*

Tsai, W. T., Zualkernan, I. A., and Kirani. S. 1992a. Pragmatic testing methods for expert systems. Technical Report TR 92-02, Computer Science Dept., Univ. of Minnesota, Minneapolis, MN - 55455.

Tsai, W. T., Kirani, S., and Zualkernan, I. A. 1992b. Comparative evaluation of expert system testing methods. In *Proc. IEEE the 4th Int'l. Conf. Tools for AI*, 334–341.

Tsai, W. T., Kirani, S., and Zualkernan, I. A. 1992c. Testing expert systems using conventional techniques. *Proc. the 16th Annual Computer Software & Appl. Conf.*, 320–325.

Tsai, W. T. and Zualkernan, I. A. 1990. Towards a unified framework for testing expert systems. *Proc. of Software Eng. & Knowledge Eng..*

Weyuker, E. J. and Jeng, B. C. 1991. Analyzing partition testing strategies. *IEEE Trans. Software Eng.*, 17 (7):703-711.

Yu, V. L. et al. 1984. An evaluation of MYCIN's advice. In B. G. Buchanan and E. H. Shortliffe, editors, *Rule-Based Expert Systems*, 589–596. Addison-Wesley.

Zualkernan, I. A., Tsai, W. T., and Kirani, S. 1992. Testing expert systems using conventional techniques. In *Proc. the 16th Annual Computer Software & Appl. Conf.*, 320–325.

Zualkernan, I. A. 1991. Representational breakdown in expert system development. Ph.D. diss., Dept. of Computer Science, Univ. of Minnesota.