High Quality Behavioral Verification Using Statistical Stopping Criteria

Amjad Hajjar[1], Tom Chen[1], Isabelle Munn[2], Anneliese Andrews[2], Maria Bjorkman[2]
[1]Department of Electrical and Computer Engineering
[2]Department of Computer Science
Colorado State University, Fort Collins, CO 80523
Email: chen@engr.colostate.edu

Abstract

In order to improve the efficiency of behavioral model verification, it is important to determine the points of deminishing return for a given verification strategy. This paper compares the existing stopping rules and presents a new stopping rule based on static Bayesian technique. The new stopping rule was applied to verifying 14 complex VHDL models. We used the figure of merit to compare the efficiency of the stopping rules. The results in terms of coverage and verification time were shown to consistantly outperform existing stopping rules.

Keywords: Behavioral Model Verification, VHDL, Statistical Stopping Rules.

1. Introduction

It is widely believed that the quality of a behavioral model is correlated to the obtained coverage measure during its verification process [1]. However, measuring coverage is just a small part of ensuring the behavioral model meet the desired quality goal. A more important question is how to increase the coverage during verification to a certain level with a given time-to-market constraint. Current methods to achieve the desired quality goal use brute force approaches where billions of test cases (patterns) were applied without knowing the effectiveness of the techniques used to generate these test cases (patterns) [4, 6, 9, 11, 12, 13].

For a given product, the correct set of testing techniques is often known. Time spent on verifying a behavioral model using a given testing technique should be fruitful, i.e. time spent with diminishing return should be avoided. Therefore, the question is how to determine the optimal stopping points for switching from one testing technique to another. There are existing statistical models used in software engineering to determine the proper stopping points for software testing [7, 8, 9, 10, 11, 12, 13]. These models assume certain statistical behavior with regard to coverage. The statistical behaviors assumed by the statistical stopping rules used for software testing include Binomial, Poisson, and Logarithmic Series distributions. However, it has been shown, theoretically and experimentally [25], that the underlining assumptions about the statistical behavior of coverage are invalid for behavioral models. We have presented a more accurate statistical behavior based on verification results of 14 VHDL models applying one million test patterns to each model. The resulting statistical behavior is different from the assumptions made in the existing statistical models. It takes into account many unique issues in VHDL model verification, such as clumping, i.e. dependency of covering one branch on covering other branches.

This paper proposes a statistical stopping rule for behavioral model verification based on the new statistical behavior presented in [25]. Section 2 gives a brief review on the existing stopping rules used in behavioral models and a discussion about the assumptions presumed in each stopping rule. Section 3 presents the mathematical derivation of the proposed stopping rule. Section 4 illustrates a comparison study of all existing stopping rules and the proposed one conducted on 14 behavioral models. Conclusions are given in Section 5. Throughout our discussion in this paper, we use the notation that the branch coverage is a good measure of quality for behavioral model verification [14], and the goal of behavioral model verification is to achieve the highest possible branch coverage within a given time constraint. Testing criteria other than branch coverage can be used without loosing the applicability of the results presented in this paper.

2. Existing Statistical Models

Almost all of the existing statistical models used to determine stopping points stem from research results in software engineering. During the past 30 years, many models have been proposed assessing the reliability measurements of software systems [7, 8]. These models help designers evaluate, predict, and improve the quality of their software systems. Goel [8] classified some of the existing software reliability models according to their failure processes [16, 15]. However, software reliability models aim at estimating the remaining faults in a given software program. Hence, the direct use of such models in estimating the number of remaining uncovered branches in a behavioral model is not beneficial since we know exactly how many branches are left. Instead, one could slightly modify the estimation process to focus on the expected number of faults, or coverage items in the case of behavioral model verification, within the next unit of testing time. Unfortunately, all the existing software reliability models assume that failures occur one at a time. Based on this assumption, expectations of the times between failures are carried on. In observing new coverage items in a behavioral model, branches are typically covered in clumps. Besides, the assumption made in the software reliability model that failures are independent of each other in the program makes the use of such models ineffective and inaccurate.

2.1. Confidence Based Models (Howdens' Models)

This approach takes advantage of hypothesis testing in determining the saturation of the software failures [10]. A null hypothesis H_0 is performed and later examined experimentally based on an assumed probability distribution for the failures. We assume that H_0 is false and observe the outcome as a failure. Suppose that the outcome has a probability less than or equal to B, then we are at least $1 - B$ confident that H_0 is true. Similarly, if the failures for the next period of testing time have the same equal probability of at least B to occur, then for the next N testing cycles, we have a confidence of at least C that no failures will happen, where

$$C = 1 - (1 - B)^N \tag{1}$$

If we experience a failure during the next N tests, then we continue testing and examine the hypothesis again. Otherwise, we stop testing at the end of the N tests.

To apply Howden's model to the process of HDL verification, we first need to treat failures as interruptions, where an interruption is an incident where one or more new part of the model are exercised. Using branch coverage as a test criterion, an interruption, therefore, indicates one or more new branches are covered. We set a probability for the interruption rate B and choose an upper-bound level of confidence C. Experimentally, we do not examine the hypothesis unless the interruption rate becomes smaller than the preset value B. If the interruption rate becomes smaller than B, we calculate the number of test patterns needed to have at least C confidence of not having any new branch in the next N test patterns and run them. If an interruption occurs, we continue examining the hypothesis until we prove it and then stop.

In this approach, we assume that coverage items, or indeed interruptions, are independent and have equal probabilities of being covered. One could use the fact that the rate of interruption is decreasing and that we assume no interruptions will occur in the next N test cases; then the expected probability of interruptions will be:

$$B_t = \left(\frac{B}{t + T}\right) \tag{2}$$

where, T is the last checked point in testing. This will then change the confidence equation as follows:

$$C = 1 - \prod_{t=1}^{N} \left(1 - \frac{B}{t + T}\right) \tag{3}$$

Howden's formulae can be implemented by first estimating B' by taking the cumulative sum of successes up to test case t and dividing by the number of test cases executed. The calculation of B' is repeated on the next test case until it becomes less than the predetermined level B. The number of test cases N needed to gain confidence C is calculated from either the above formulae, and N test cases are executed. If no new coverage is gained during the execution of the test strategy, it indicates that the current verification strategy is no longer effective. The stopping point is $t + N$. If new coverage is gained at test case k, go back to step 1 and repeat the steps for $t = t + k$ times.

In Howden's model, the assumption that failures or interruptions have a given probability B independently is erroneous. Branches in an HDL model, as we know, are strongly dependent of each other. In fact, we can classify some branches to be dominant to other branches where it is impossible to cover the lower level ones without covering their dominators. Moreover, the sizes of the interruptions are not modeled in this study making the decision of continuing or stopping the testing process inaccurate. Lastly, this work doesn't incorporate the cost of testing or releasing the product, and the goal of testing in the first place is not only having a high quality product but also minimizing the testing costs.

2.2. Binary Markov Model

Sanping Chen and Shirley Mills developed a statistical Markov process model [11]. The probabilistic distribution assumptions of the model are the same as Howden's except that failures are statistically dependent with a certain unknown correlation constant ρ. Again, if interruptions are correlated, then the probability of having no interruptions in the next N test cases is then given by:

$$p(0|N, B, \rho) = (1 - B)(1 - B + \rho B)^{N-1} \tag{4}$$

which makes the confidence $C = 1 - p(0|N, B, \rho)$. If we set ρ to be zero, interruptions are independent; then we get the same model as Howden's. The implementation of this stopping rule is similar to that of Howden's except in the calculation N for the confidence.

The basic assumption that interruptions have this simple probability distribution is not well understood or clearly proven. Furthermore, the value of ρ in this model is unknown, and authors experimentally assumed different values ranging from 0 to 0.9 and obtained different results. Thus, this correlation needs to be determined.

2.3. Testing Cost Based Models (Dalal-Mallows)

Dalal and Mallows [12] assumed a loss function associated with the testing and releasing of software programs. If, up to time t, there are $K(t)$ number of bugs in the model, then $aK(t)$ is the cost of fixing these bugs while testing, and $b(N - K(t))$ is the cost of fixing the remaining bugs in the field after releasing the product for some constant a and b. N is the expected number of total bugs in the program. Finally, there is also a fixed increasing cost of the testing setup and running of $f(t)$. Thus, the loss function is defined as:

$$L(t, N) = f(t) + aK(t) - b(N - K(t)) \tag{5}$$

Under these cost assumptions, the decision is to stop when the loss function is not decreasing. That reduces the loss function to a reward function defined as $cK(t) - f(t)$, c=b-a, since N is a fixed number, yet unknown. Testing stops when the expected reward function is no longer increasing. Now, $K(t)$ is a random process distributed as a non-homogeneous Poisson process with increments $\lambda g(t)$. That simplifies the stopping rule to the following:

$$\frac{f'(t)\ G(t)}{c\ g(t)} \geq K(t) \tag{6}$$

If we assume $g(t)$ to be exponential, and the cost function $f(t)$ is linear with time, then the decision to stop is when:

$$\frac{f}{\mu c}(e^{\mu t} - 1) \geq K(t) \tag{7}$$

where μ is maximum likelihood estimate of the history that $K(t)$ is Poisson with mean $\lambda(1 - e^{\mu t})$. μ is the solution of the following equation:

$$\frac{\mu\ (e^{\mu t} - 1)}{e^{\mu t} - 1 - \mu t} = \frac{K(t)}{S(t)} \tag{8}$$

$S(t)$ is the sum of all failures' life times up to test case t. This model incorporated the cost of testing and gave reasonable assumption to failures occurring in a certain program. However, applying this model to coverage items as failures suffers the independency problem of the Poisson process, where the times between failures are independent, although the parameters of the distributions are modified by the time and the cost constants. The model also implies the assumption of not having clumped failures, which reduces the efficiency of the model when applying it to branch coverage estimation. Finally, this stopping rule diverged in some testing phases of some VHDL models. The reason is that the mathematical constructions used by this rule theoretically allow major changes in the internal constants values during the testing process, which ultimately make the rule diverge.

2.4. Compound Poisson Stopping Rule

This model was the first attempt in modeling the branch coverage process of VHDL circuits utilizing the benefits of the cost modeling of Dalal and Mallows [12] and solving the clumps phenomenon of branches, explained in the previous section, being covered in the testing process [17]. This model uses the empirical Bayesian principles for the compounded counting process. It was previously introduced as a software reliability model for failure estimation in 1992 [13] and later modified to incorporate the cost modeling proposed by Dalal and Mallows in 1995 [18, 19]. The model was recently formulated to model the branch coverage process in VHDL models [24, 23].

The idea is to compound potentially two probability distributions, for both the number of interruptions and the size of interruptions. The resulting compound distribution is assumed to be the probability distribution function of the total number of failures, or coverage items, at a certain testing time point. The parameters of the distributions are also assumed to be random variables calculated empirically based on the well-known Bayesian estimation.

For modeling the branch coverage process for HDL models, it is assumed that the number of interruptions over the time, $N(t)$, is a Poisson process with mean λ, and the size of each given interruption, W_i, is distributed as a Logarithmic Series Distribution (LSD). The resulting compound distribution for the total number of failures, the sum of the sizes, is also known as a Negative Binomial distribution (NBD), if the Poisson parameter λ is set to $-k\ln(1-\theta)$.

The expected value of coverage in the next unit of testing time when testing is observed up to time t is estimated as:

$$E(X) = k\ \frac{\alpha + x}{\beta + k} \tag{9}$$

where x is the number of branches covered up to time t, α and β are the constants of the *priori* distribution of the parameter of the LSD, and k is set such that the compound distribution becomes a Negative Binomial. So:

$$e^{\frac{\lambda}{k}} = 1 + \frac{\alpha + x}{\beta + k} \tag{10}$$

is a nonlinear equation for k that can be solved using the Newton method. λ is the interruption rate up to time t.

This expected coverage in the next unit of time is then used in the cost model proposed in [12] to decide whether to continue or to stop testing with the current testing strategy. If the expected cost of testing for the next unit time is more than the expected cost of stopping, then the decision is to stop, and vise versa. This decision can be formulated as:

$$aE(X) < bE(X) + c \qquad (11)$$

where a is the cost of one coverage item as yet uncovered, c is a fixed cost of one unit of testing time, and b is the variable cost of testing one uncovered branch. In other words, the decision to stop is when $E(X) < d$, for $d = \frac{c}{a-b}$, and checking this decision is to be made sequentially after applying each test pattern or case.

This stopping rule models the clumps of the coverage items in a statistical manner updating the assumed probability distribution parameters in every test case based on the testing history. However, since the interruption sizes are distributed as LSD, there should be at least one new coverage item per interruption covered. The problem involved was that after a short period of time, the coverage activity died for most of the test patterns applied sequentially, and for few patterns after that, one or more branches were covered. To overcome this problem, the outcomes of the simulation were packed into groups as if the time scale of the testing process were compact, and the packed branch coverage was applied to the stopping rule for the stopping decision.

2.5. The Sequential Sampling Models

All the previously discussed stopping rules assume that the failures or interruptions are random processes according to a given probability distribution. Another software reliability technique that doesn't involve any assumptions on the probability distributions for the failure process was presented in [22]. Recently, the technique is applied to VHDL models in determining the stopping points for a given testing history of branch coverage [21]. The model evaluates the stopping decision based on three key factors: the discrimination ratio (γ), the supplier risk (α), and the consumer risk (β). γ represents the maximum number of input test patterns accepted that don't yield any more coverage. The supplier risk is the probability of falsely believing that the testing process should be stopped; the consumer risk is the probability of falsely believing that the testing process should continue. If the number of cumulative coverage at time t is $X(t)$, then the testing process should be stopped when:

$$X(t) \leq \frac{ln\left(\frac{1-\beta}{\alpha}\right) - t\ ln(\gamma)}{1-\gamma} \qquad (12)$$

Figure 1 shows the decision boundary of the sequential sampling model when applied to one of the behavioral models at some testing phases. The stopping decision is made when the boundary line intersects with the increasing coverage at a certain point of time as illustrated in Figure 1. The stopping decision strongly depends on the value of γ much more than α and β, and we can see clearly that this decision doesn't incorporate any cost model of the testing process. In [21], we modified γ with respect to testing strategies such that if higher coverage were achieved in the previous test strategy, γ is increased in the current test strategy in order to decrease the expectation of achieving more coverage in the current strategy. The new value of γ, therefore, becomes:

$$\gamma^{+} = \gamma\ ln(\Delta) \qquad (13)$$

where Δ is the coverage increase achieved in the previous test strategy. The value of γ remains the same if $\Delta \leq e$.

This type of statistical modeling doesn't use any priori probability distribution for the data provided. This is one reason the sequential sampling models are widely used in many testing areas. However, the cost of testing is not modeled in making the stopping decision, and the stopping point determined by the sequential sampling model is very sensitive to the γ value chosen during the testing process.

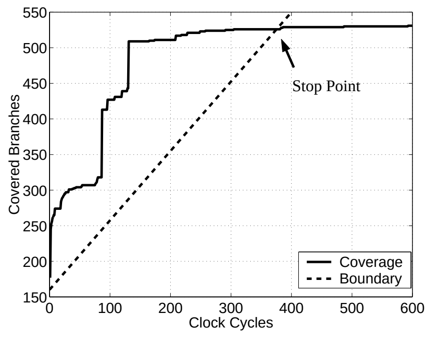

Figure 1. Sequential Sampling Decision Line

3. The Bayesian Based Model

The branch coverage process, X_t, can be decomposed into two random variables: interruptions N_t, where one or more new branches are covered at time t, and sizes of the interruptions, W_t, given that there is an interruption at the same time. Interruptions, N_ts, are strongly dependent of each other, governed by a correlation function. The sizes of interruptions, however, are not that strongly dependent especially after a long period of testing where achieving new coverage will be more and more difficult. Thus, the random variables, W_ts, are assumed statistically independent.

PMF extraction experiment (also known as *distribution harvesting*) was conducted to estimate the best fitted distribution function to the histograms of the actual interruption sizes, W_t, at every discrete time t based on verification results of 14 VHDL models applying one million test patterns for each model. The best fitted distribution to W_t is found to be a non-homogeneous Poisson process:

$$W_t \sim e^{\beta_t} \frac{\beta_t^{w-1}}{(w-1)!} \tag{14}$$

For the number of interruptions, N_t, the probability of having an interruption during the testing process is estimated as $p(t)$ for every discrete time t. This $p(t)$ function is further decomposed into a shape function and an amplitude value, $p(t) = \zeta\, f(t)$, where ζ is a constant value. The shape function, however, is statically chosen such that it best describes the power of increments of the interruptions N_t. This choice is made based on a *shape fitting* experiment, and found to be a logarithmic polynomial shape [25]:

$$p(t) = \zeta \ln \frac{t}{t-1} \tag{15}$$

Using the distribution and the correlation for the branch coverage process, X_t, presented in [25], we derive the statistical model for the branch coverage for a given history of testing simulation. Let $x_t, t = 1, 2, 3, \ldots, T$, be the cumulative number of actual branches covered from the beginning of the testing simulation up to time t. Let n_T be the number of interruptions up to time T. Let W_t be the random variables of the sizes of the interruptions $t = 1, 2, \ldots, n_T$. We have:

$$X_T = \sum_{t=1}^{n_T} W_t \tag{16}$$

The conditional distributions for the sizes of interruptions, W_ts, given that there will be interruptions at times t, and the occurrences of interruptions, D_t, have the following distributions:

$$W_t|\{D_t, \beta_t\} \sim \frac{e^{-\beta_t}\, \beta_t^{w_t-1}}{(w_t-1)!}\, d_t \tag{17}$$

$$D_t \sim p(t)\, d(t) + q(t)\, (1 - d(t)) \tag{18}$$

In order to start the Bayesian analysis, the Likelihood function of β_t should first be estimated from the data of the verification history, $\vec{w}$ and $\vec{d}$. Given the distribution functions of Equation 18, the Likelihood function of β_t can be estimated to be:

$$L(\beta_t|\vec{w}, \vec{d}) \propto e^{-\beta\, G(t)} \times \beta^{x_t - n_t} \tag{19}$$

where $\beta_t = \beta\, g(t)$ for some constant β and a decreasing function $g(t)$, $G(t) = \sum_{j:d_j=1} g(j)$, x_t is the coverage up to time t, and n_t is the number of interruptions up to time t.

From the Likelihood of equation (19), we then estimate the constant β, that is assumed to be distributed as $\Gamma(r, \gamma)$, using the Bayesian method as:

$$\hat{\beta} = \frac{r + x_t - n_t}{\gamma + G(t)} \tag{20}$$

for some constants γ and r describing the prior distribution of β. Both constants, γ and r, can be set to 1 since they are initial values that will immediately be modified through the sequential updates of the Bayesian process.

Since W_t is a shifted Poisson random process, the expected value of W_t given β_t is:

$$E\{W_t|\beta_t\} = \beta_t + 1 \tag{21}$$

Now, we expect this conditional expectation of W_t over the values of β_t to get the absolute expected value of W_t at any time t, and we get:

$$E\{W_t|\vec{x}\} = 1 + \hat{\beta}\, g(t) \tag{22}$$

And by utilizing equation (20), the expected size of interruption at certain time t is given by:

$$E\{W_t|\vec{x}\} = 1 + \frac{r + x_t - n_t}{\gamma + G(t)}\, g(t) \tag{23}$$

From this expectation, we can expect the total number of branches X_t to be covered at time $t = T+1$ for a given history of testing up to time T. At the exact time T, we know that we have covered x_T branches. The expectation of the size of the interruption at time $T+1$ is $E\{W_{T+1}|\vec{x}\}$ given that there will be an interruption at time $T+1$. Now, our expectation that there will be an interruption at time $T+1$ is the correlation function $p(T+1)$. Thus, the expected value of X_t at time $T+1$ given the history of testing up to time T is:

$$\begin{aligned} E\{X_t|\vec{x}\} &= x_T + E\{W_{T+1}|\vec{x}\} \times p(T+1) && (24)\\ &= x_T + \left(1 + \frac{r + x_T - n_T}{\gamma + G(T)} g(T+1)\right) && (25)\\ &\times \zeta\, f(T+1) \end{aligned}$$

where ζ can be set such that $p(t) = 1$ at time $t = 1$:

$$\zeta = \frac{1}{ln(2)} = 1.44 \tag{26}$$

We use the stopping criterion similar to that in [23, 12] and not only for the next unit of testing time t but also for the remaining expected coverage (i.e. $E\{X_t\}$) utilizing the proposed expectation in (23). More details on the derivation of the statistical models in this section can be found in [26].

4. Experimental Results and Comparisons

The experiment setup included a functional test phases followed by 4 different random test phases. Each stopping rule was applied to all the phases to decide when to stop and switch to the next test strategy. The abbreviation of each stopping rule used is shown in Table 1. 14 different VHDL models were used in our experiments. The characteristics of these 14 VHDL models are shown in Table 2. Table 3 shows the results of applying all the existing stopping rules as well as the proposed static Bayesian stopping rule to verifying 14 VHDL models listed in Table 2. For each entry in Table 3, the number on the top is the number of branches covered during the verification process, and the number below is the total number of test patterns applied during the verification.

Orig:	Original Run without stopping
SS1:	Sequential Sampling Fixed γ
SS2:	Sequential Sampling Variable γ
HW1:	Howden First Formula
HW2:	Howden Second Formula
BM:	Binary Markov Model
DL:	Dalal-Mallows Cost Rule
CP:	Compound Poisson Rule
SB:	Proposed Static Bayesian Rule

Table 1. Stopping Rules

Model	LOC	P	C	B	DCFG
Sys7	3785	62	7	591	7
8251	3113	3	12	207	9
B01	1880	7	9	373	8
B04	4657	42	15	251	3
B05	5015	46	19	302	6
B06	4667	39	16	225	6
B07	4710	39	16	225	6
B08	4949	52	15	296	3
B09	4963	46	19	302	6
B10	4777	39	16	225	6
B11	4752	42	15	251	3
B12	4973	46	19	302	6
B14	5498	53	21	399	6
B15	5770	66	21	470	6

LOC:	Lines of VHDL Code
P:	Number of Process Blocks
C:	Input Control Bits
B:	Number of Branches
DCFG:	Depth of Control Flow Graph

Table 2. VHDL Models Characteristics

To compare the proposed static Bayesian stopping rule with the existing stopping rules, we use the *Figure of Merit* function, fm, defined as:

$$fm \;=\; cov - \alpha\, tt \tag{27}$$

where cov is the branch coverage percentage reached after applying the stopping rule, and tt is the number of testing units (clock cycles) used. The fm function gives the indication of the efficiency of a given stopping rule when applied to a behavioral model. However, α in equation (27) should be set such that it reflects the cost/benefit ratio of verification versus quality. In our comparison analysis, we select three different values for α to show the effectiveness of the proposed stopping rule. The fm results for the different stopping rules and for the three selected α values are shown in Table 4 in a normalized form.

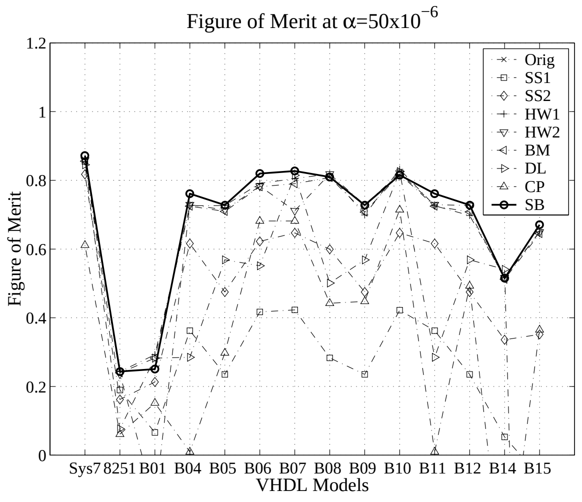

Figure 2. fm **at** $\alpha = 50\mu$

Table 4 shows that the figure of merite values of the proposed SB stopping rule are of the highest values for most the examined behavioral models when α is set to 10μ, which indicates the cost of continuing testing is low. We notice that for $\alpha = 10\mu$ there are 30 cases out of 126 in the Table that yielded better fm values than that of SB. When α is set to 50μ, which indicates the cost of continuing testing is medium, only 9 cases yielded better fm values than that of SB. Figure 2 shows the figure of merit lines when α is set to 50μ. When the α value is chosen higher, 12 cases yielded better fm values than that of SB. On the average across the behavioral models, however, the highest normalized fm values were yielded by the SB stopping rule when α is set to 50μ and 100μ. The next best normalized fm is 0.99 for $\alpha = 50\mu$, and 0.97 for $\alpha = 100\mu$ both yielded by the HW1 stopping rule. When $\alpha = 10\mu$, we found slightly better fm values with BM, HW1, and HW2 methods. We notice that these three stopping rules use the confidence-based criterion in the stopping decision, and they yielded higher fm values when the cost of testing is almost negligible. That shows the

Model	Orig	SS1	SS2	HW1	HW2	BM	DL	CP	SB
Sys7	568	536	538	536	536	536	536	547	535
	54283	1039	1858	927	969	1025	1235	6287	661
8251	161	73	73	79	79	81	75	112	74
	81500	3259	3812	2769	2906	3033	5712	9600	2275
B01	200	177	142	128	155	128	128	135	128
	80000	8169	3352	1010	11084	1211	1211	4200	1854
B04	223	220	218	206	214	214	219	217	199
	80000	10282	5047	1894	2468	2557	11755	17100	631
B05	259	234	251	251	251	251	252	253	232
	80000	10795	7122	2092	2343	2431	5318	10800	808
B06	210	192	192	192	192	192	204	204	192
	80000	8725	4618	1240	1407	1439	7110	4500	673
B07	210	196	198	196	204	196	196	204	195
	80000	8963	4660	1322	3904	1621	1132	4500	789
B08	274	268	268	263	263	273	273	273	273
	80000	12447	6122	1392	1405	2283	8427	9600	2249
B09	260	234	251	234	251	251	252	253	232
	80000	10795	7122	1512	2053	2470	5324	7800	809
B10	210	197	198	204	204	204	196	208	208
	80000	9068	4660	1488	1711	1781	915	4200	2181
B11	223	220	218	206	214	214	219	217	199
	80000	10282	5047	1894	2468	2557	11755	17100	631
B12	259	234	251	234	251	251	252	253	232
	80000	10795	7122	1545	2085	2462	5318	6900	808
B14	257	248	248	244	244	244	248	253	245
	80000	11367	5712	1892	1900	1991	1618	21000	1982
B15	415	351	351	350	350	350	418	383	364
	80000	16190	7906	1892	1900	1991	80002	9000	2080

Table 3. Stopping Rules' Coverage Results

effectiveness of the confidence-based criterion in the stopping decision when the testing cost is very low. However, the proposed SB stopping rule shows robustness for a much wider range of testing costs. Table 5 lists the normalized averaged fm values for the three selected α's.

5. Conclusion

We have presented a more efficient stopping rule to improve the efficiency of behavioral model verification. The stopping rule used a static Bayesian methodology utilizing experimental statistical behavior to estimate the "good" point to switch or stop the testing process. The proposed stopping rule incorporates the cost model criterion in the stopping decision. Results show that the proposed stopping rule outperforms the existing stopping rules in terms of efficiency measured by a figure of merit function. More investigation and better choices of stopping criteria could be used for better performance.

References

[1] B. Barrera, "Code coverage analysis-essential to a safe design", Electronic Engineering, pp 41-44, November 1998

[2] T. Boyle, M. Abrahams, "Measuring ASIC test coverage", Electronic Product Design, pp 41-42, October 1996

[3] M. Abrahams, S. Riches, "Optimize ASIC test suites using code-coverage analysis", EDN, pp 149-152, May 1998

[4] B. Dickinson, S. Shaw, "Software techniques applied to VHDL design", New Electronics, n9, pp 63-65, May 1995

[5] J. Lipman, "Covering your HDL chip-design bets", EDN, pp 65-70, October 1998

[6] J. Gately, "Verifying a million gate processor", Integrated System Design, pp 19-24, 1997

[7] S. Gokhale, K. Trivedi, "Log-logistic software reliability growth model", University of California, pp 34-41, 1998

[8] A. Goel, "Software reliability models: assumptions, limitations, and applicability", Software Eng., v SE-11, n 12, pp 1411-1423, Dec. 1985

[9] W. Howden, "Confidence-based reliability and statistical coverage estimation", ISSRE'97, pp 283-291, Nov. 1997

[10] W. Howden, "Systems testing and statistical test data coverage", COMPSAC, pp 500-505, Aug. 1997

[11] S. Chen, S. Mills, "A binary Markov process model for random testing", Software Eng., v22, n3, pp 218-223, 1996

[12] S. Dalal, C. Mallows, "When should one stop testing software?", Journal of the Ameri can Statistical Association, v 83, n 403, pp 872-879, September 1998

Model	Orig	SS1	SS2	HW1	HW2	BM	DL	CP	SB
Sys7	0.47	1.00	0.99	1.00	1.00	1.00	1.00	0.96	1.00
8251	ve	0.96	0.94	1.06	1.05	1.08	0.91	1.33	1.00
B01	ve	1.21	1.07	1.03	0.94	1.02	1.02	0.99	1.00
B04	0.11	0.98	1.04	1.02	1.05	1.05	0.96	0.88	1.00
B05	0.08	0.88	1.00	1.07	1.06	1.06	1.03	0.96	1.00
B06	0.16	0.90	0.95	0.99	0.99	0.99	0.99	1.02	1.00
B07	0.16	0.91	0.97	1.00	1.01	1.00	1.00	1.00	1.00
B08	0.14	0.87	0.94	0.97	0.97	1.00	0.93	0.92	1.00
B09	0.08	0.88	1.00	1.00	1.07	1.06	1.03	1.00	1.00
B10	0.15	0.87	0.92	0.99	0.99	0.98	0.95	0.98	1.00
B11	0.11	0.98	1.04	1.02	1.05	1.05	0.96	0.88	1.00
B12	0.08	0.88	1.00	1.00	1.07	1.06	1.03	1.01	1.00
B14	ve	0.85	0.95	1.00	1.00	1.00	1.02	0.71	1.00
B15	0.11	0.78	0.89	0.96	0.96	0.96	0.12	0.96	1.00

Normalized fm for $\alpha = 10\mu$

Model	Orig	SS1	SS2	HW1	HW2	BM	DL	CP	SB
Sys7	ve	0.98	0.94	0.99	0.98	0.98	0.97	0.70	1.00
8251	ve	0.78	0.66	1.00	0.97	0.98	0.31	0.25	1.00
B01	ve	0.26	0.85	1.17	ve	1.13	1.13	0.61	1.00
B04	ve	0.48	0.81	0.95	0.96	0.95	0.37	0.01	1.00
B05	ve	0.32	0.65	1.00	0.98	0.97	0.78	0.41	1.00
B06	ve	0.51	0.76	0.97	0.96	0.95	0.67	0.83	1.00
B07	ve	0.51	0.78	0.97	0.86	0.96	0.98	0.82	1.00
B08	ve	0.35	0.74	1.01	1.01	1.00	0.62	0.55	1.00
B09	ve	0.32	0.65	0.96	1.00	0.97	0.78	0.62	1.00
B10	ve	0.52	0.79	1.02	1.01	1.00	1.01	0.88	1.00
B11	ve	0.48	0.81	0.95	0.96	0.95	0.37	0.01	1.00
B12	ve	0.32	0.65	0.96	1.00	0.97	0.78	0.68	1.00
B14	ve	0.10	0.65	1.00	1.00	0.99	1.05	ve	1.00
B15	ve	ve	0.52	0.97	0.97	0.96	ve	0.54	1.00

Normalized fm for $\alpha = 50\mu$

Model	Orig	SS1	SS2	HW1	HW2	BM	DL	CP	SB
Sys7	ve	0.96	0.86	0.97	0.97	0.96	0.93	0.35	1.00
8251	ve	0.21	ve	0.81	0.70	0.68	ve	ve	1.00
B01	ve	ve	0.29	1.53	ve	1.41	1.41	ve	1.00
B04	ve	ve	0.50	0.87	0.83	0.82	ve	ve	1.00
B05	ve	ve	0.17	0.90	0.87	0.86	0.44	ve	1.00
B06	ve	ve	0.50	0.93	0.91	0.90	0.25	0.58	1.00
B07	ve	ve	0.53	0.94	0.66	0.90	0.96	0.58	1.00
B08	ve	ve	0.42	1.07	1.07	1.00	0.11	ve	1.00
B09	ve	ve	0.17	0.91	0.91	0.85	0.44	0.08	1.00
B10	ve	ve	0.59	1.07	1.04	1.03	1.10	0.71	1.00
B11	ve	ve	0.50	0.87	0.83	0.82	ve	ve	1.00
B12	ve	ve	0.17	0.90	0.91	0.85	0.44	0.21	1.00
B14	ve	ve	0.12	1.02	1.01	0.99	1.11	ve	1.00
B15	ve	ve	ve	0.98	0.98	0.96	ve	ve	1.00

Normalized fm for $\alpha = 100\mu$
ve**:** means negative ratio

Table 4. Normalized fm Values

[13] M. Sahinoglu, “Compound Poisson software reliability model”, Software Eng., v18, n7, pp 624-630, July 1992

[14] A. von Mayrhauser, et. al., “On choosing test criteria for behavioral level hardware design verification”, HLDVT’00, Berkeley, CA, Nov. 2000.

[15] D. Mills, “On the statistical validation of computer programs”, IBM FSD, Report FSC-72-6015, 1972.

[16] J. Musa, “A theory of software reliability and its application”, Software Eng. SE-1(3), pp 312-27, 1975

Rule	$\alpha = 10\mu$	$\alpha = 50\mu$	$\alpha = 100\mu$
Orig	0.10	-4.5	-11.4
SS1	0.92	0.43	-0.30
SS2	0.99	0.74	0.39
HW1	1.01	0.99	0.97
HW2	1.01	0.93	0.82
BM	1.03	0.97	0.92
DL	0.93	0.38	-0.44
CP	0.96	0.49	-0.25
SB	1.00	1.00	1.00

Table 5. Normalized Averaged fm Values

[17] M. Sahinoglu, et. al., “On the efficiency of a compound Poisson stopping rule for mixed strategy testing”, IEEE Aerospace, Track#7, March 1999

[18] M. Sahinoglu, U. Can, “Alternative parameter estimation methods for the compound Poisson software reliability model with clustered failure data”, Software Testing, Veri., and Rel., v7, pp 35-57, 1997

[19] P. Randolph, M. Sahinoglu, “A stopping rule for a compound Poisson random variable”, Applied Stochastic Models and Data Analysis, v11, pp 135-143, 1995

[20] R. Ferguson, B. Korel, “Generating test data for distributed software using the chaining approach”, Information and Software Tech., v38, n5, pp 343-353, 1996

[21] T. Chen, I. Munn, A. von Mayrhauser, A. Hajjar, ”Efficient verification of behavioral models using the sequential sampling technique”, VLSI’99, Brazil, 1999

[22] J. Musa, *et. al.*, Software reliability: measurement, prediction, application, McGraw-Hill, 1987.

[23] T. Chen, et. al., “Achieving the quality for behavioral models with minimum effort”, ISQED’00, pp , March 2000

[24] T. Chen, M. Sahinoglu, A. von Mayrhauser, A. Hajjar, Ch. Anderson, “How much testing is enough? Applying stopping rules to behavioral model testing”, HASE’99, Washington, DC, pp 249-256, Nov. 1999

[25] A. Hajjar, T. Chen, A. von Mayrhauser, “On statistical Behavior of Branch Coverage in Testing Behavioral VHDL Models”, IEEE HLDVT’00, Nov 2000.

[26] A. Hajjar, “Bayesian based stopping rules for behavioral VHDL verification”, PhD Dissertation, Electrical Eng. Dept., Colorado State Univ., 2000