ISSN-2230-9659

To Minimize Cost Subject to the Expected Number of Failures Remaining Constraint

Sonia
Research Scholar, Singhania University,
Pacheri Bari, Jhunjhunu, Rajasthan

INTRODUCTION

Growth in Mathematics and engineering technology has led to production of Mathematics for highly complex situations occurring in industry, scientific research, defense and day to day life. The computer revolution is fueled by an ever more rapid technological advancement. Today, computer hardware and Mathematics permeates our modern society. Computers are embedded in wristwatches, telephones, home appliances, buildings, automobiles, and aircraft. Science and technology demand high-performance hardware and high-quality Mathematics for making improvements and breakthroughs. We can look at virtually any industry - automotive, avionics, oil, telecommunications, banking, semi-conductors, pharmaceuticals - all these industries are highly dependent on computers for their basic functioning. When the requirements for and dependencies on computers increase, the possibility of cries from computer failures also increase. It is always desirable to remove a substantial number of faults from the Mathematics. In fact

ISSN-2230-9659

the reliability of the Mathematics is directly proportional to the number of faults removed. Hence the problem of maximization of Mathematics reliability is identical to that of maximization of fault removal. At the same time testing resource are not unlimited, and they need to be judiciously used.

In focusing on error prevention for reliability, we need to identify and measure the quality attributes applicable at different life cycle phases. As discussed previously, we need to specifically focus on requirements, design, implementation, and test phases.

Mathematics development process is often called Mathematics Life Cycle, because it describes the life of a Mathematics product from its conception to its implementation. Every Mathematics development process model includes system requirements as input and a delivered product as output. Many life cycle models have been proposed, based on the tasks involved in developing and maintaining Mathematics, but they all consist of the following stages and faults can be introduced during any of these stages.

Minimize Cost Subject to a Reliability Constraint

We

$$Minimize\ C(T)$$

Subject to $R(x|T) \geq R_0$

We determine a Mathematics release time T such that the expected total Mathematics cost is minimized subject to Mathematics reliability not less than a specified value, R_0.

. Theorem 4: For given C_3 , T_u , $C_{i,1}$ and $C_{i,2}$ for i=1,2,3

(1) If $f(0) > C_3$, then

ISSN-2230-9659

(l a) if $R(x|T=0) \geq R_0$, then $T^* = T_0$

(l b) if $R(x|0) < R_0 \leq R(x|T_0)$, then $T^* = T_0$

(1c) if $R(x|T_0) < R_0$, then $T^* = T_1$

(2) If $f(0) \leq C_3$, then

(2a) if $R(x|T=0) \geq R_0$, then $T^* = 0$

(2b) if $R(x|0) < R_0$, then $T^* = T_1$

where T_0 is the solution from Theorem 3 and T_1 is the solution of $R(x|T_1) = R_0$ Statement (1) notes that when the current amount of debugging does not minimize the total cost of the Mathematics system and further debugging should be done, then the question is at what time should the debugging be stopped and the Mathematics released. Statement (l a) notes that if the current amount of debugging has met the reliability constraint then further debugging should be done until time T_0 where T_0 is defined above. Statement (l b) notes that if the current amount of debugging has not met the reliability constraint but the reliability constraint will be met at a time T_1 , which is less than time T_0, then the program should be debugged until time T_0. Statement (1c) notes that if the current amount of debugging has not met the reliability constraint and the reliability constraint will be met at a time T_1, which is greater than T_0, then the program should be debugged until time T_1.Similarly, statement (2) notes that the current amount of debugging has already minimized the Mathematics system cost and that any further debugging will increase the total cost. This would imply that no further debugging should be done, but since there is a reliability constraint it must be checked to see if it has been met. Statement (2a) notes

ISSN-2230-9659

that the current amount of debugging has met the reliability constraint, and therefore, no further debugging should be done and the Mathematics should be released. Statement (2b) notes that the current amount of debugging does not meet the reliability constraint and that debugging should be continued until T_1 where T_1 satisfies $R(x|T = T_1) = R_0$.

Minimize Cost Subject to the Expected Number of Failures Remaining Constraint

In the previous two sections optimal release times were determined for minimizing cost subject to no constraints and minimizing cost subject to a reliability constraint. Neither of these two sections considered the fact that there are three types of errors that determine the reliability of the system. In this section a method is presented that allows for constraining a particular type of error. Its importance is that while a program may be able to tolerate a number of minor errors it cannot tolerate critical errors. In this case a constraint can be put on the expected number of critical errors in the system before its release. It is also possible to set up constraints for each of the different types of errors independent of the other types. Consider both the expected total Mathematics system cost *C*(*T*) and the expected number of failure type *i* errors remaining in the system $\overline{m}_i(T)$, as the evaluation criteria, then the optimal release problem can be formulated as

$$Min \ C(T) \qquad (3.25)$$

$$\text{Subject to} \quad \overline{m}_i(T) \le d_i \ \text{ for all } i.$$

Note that

ISSN-2230-9659

$$\bar{m}_i(T) = m_i(\infty) - m_i(T) \quad \text{for all} \quad i$$

$$m_i(T) = \frac{ap_i}{1-\beta_i}[1 - e^{-(1-\beta_i)b_i T}]$$

$$m_i(\infty) = \frac{ap_i}{1-\beta_i}$$

Therefore, we obtain

$$\bar{m}_i(T) = \frac{ap_i}{1-\beta_i} e^{-(1-\beta_i)b_i T}$$

$$\frac{ap_i}{1-\beta_i} e^{-(1-\beta_i)b_i T} \leq d_i$$

if and only if

$$T \geq \frac{\ln[\frac{d_i(1-\beta_i)}{ap_i}]}{-(1-\beta_i)b_i} \equiv T_m \qquad (3.26)$$

MODEL PARAMETER ESTIMATION

Since the data being used is of the form specified those equations were used to estimate the parameters. Specifically, the three equations represented by eqare solved numerically by b_1, b_2, and b_3 are then put into eq. (3.22) to find the value of *a*.

The results of these calculations are

$$b_1 = 0.000242749 \qquad b_2 = 0.00029322 \qquad b_3 = 0.000304946 \qquad a = 428$$

From these estimate and the parameters given, it is possible to determine uniquely the reliability equation and Table 1.

ISSN-2230-9659

MINIMIZE COST

To find the optimal release time in this situation, it is necessary to use the relationships of the cost function is as follows:

$$\begin{aligned} C(T) = {} & 5T + 200\,m_1(T) + 50\,b_1 m_2(T) + 10\,m_3(T) \\ & + 500[m_1(100,000) - m_1(T)] \\ & + 120[m_2(100,000) - m_2(T)] \\ & + 70[m_3(100,000) - m_3(T)] \end{aligned}$$

Using eq. (3.25), we can determine T_0 as follows

$$T_0 = 2,088 \quad hours$$

$$T^* = T_0 = 2,088 \quad hours$$

MINIMIZE COST SUBJECT TO RELIABILITY CONSTRAINT

To find the optimal release time in this situation, we use the relationships of Also a reliability constraint must be specified as

$$R(10|T) \geq 0.99$$

From this, the value of T_r, is found

$$T_r = 19,045 \ hours$$

Since T_r is the max of T_0 and T_r.

$$T^* = T_r = 19,045 \ hours$$

ISSN-2230-9659

Table -1

Reliability Model Statistics

Current Debug Time (T)	Future Time (x)	Reliability over (T,T+x)	Expected No. of Errors Remaining after (T)			Expected No. Of Errors Detected During (T, T+x)			Expected No. of Errors Introduced by (T)			Expected No. of Errors Detected by (T)		
			I		III	I		II	I	II	III	I	II	III
0	5	0.526623	15	183	289	0	0	0	0	0	0	0	0	0
	10	0.277571				0	0	1						
	20	0.077311				0	1	2						
420	5	0.537454	15	178	279	0	0	0	0	1	0	0	5	10
	10	0.289098				0	0	1						
	20	0.083856				0	1	2						
480	5	0.569109	14	163	251	0	0	0	0	4	2	1	20	38
	10	0.324131				0	0	1						
	20	0.105379				0	1	1						
5760	5	0.870438	7	47	54	0	0	0	4	27	12	8	136	235
	10	0.757801				0	0	0						
	20	0.574680				0	1	0						

MINIMIZE COST SUBJECT TO THE ERRORS REMAINING CONSTRAINT

To find the optimal release time in this situation, we use the relationships of Also remaining error constraints must be specified. The remaining error constraints to be used are

$d_1 \leq 5$ $d_2 \leq 7$ $d_3 \leq 10$

From these constraints we determined

ISSN-2230-9659

$T_1 = 8,946\ hrs$ $T_2 = 13,912\ hrs$ $T_3 = 11,607\ hrs$

Since T_2 is the max of $T_0, T_1,$ and T_3

$T^* = T_2 = 13,912\ hrs$

CONCLUSION

We have discussed a model which allows for imperfect debugging and three different error types. This is done within the framework of NHPP. The three error types are categorized by the difficulty of removal and detection. Minor errors (Type 3) are easily detected and removed; major errors (Type 2) are more difficult to detect and remove; critical errors (Type 1) are very difficult to detect and remove. We also presented an SRGM which incorporates the possibility of introducing new faults (i.e. secondary faults) into a Mathematics system due to the imperfect debugging of the original faults (i.e. primary faults) in the system. These new faults are assumed to occur in a delayed sense. Further we discussed the cost model with multiple failures, imperfect debugging as well as random life cycle in which cost also includes the penalty cost.

The probability of perfect debugging can usually be increased with additional cost and, hence, it has a strong influence on total Mathematics development cost. A concept of testing level is introduced here. To achieve the lowest Mathematics cost, the management can use the proposed cost model to formulate the optimal testing level and release time problem by considering the effect of imperfect debugging. Our problem formulation and the proposed solution is useful in practice as the imperfect debugging probability can be managed by using test engineers with proper experience, by selecting testing strategy or even by including a suitable number of review staff.