Analysis of Reliability and Cost Tradeoffs in Architecture based Software Applications using a Genetic Algorithm

Poonam Panwar, Ankur Garg
Department of Computer Science & Engineering
Ambala College of Engineering & Applied Research, Ambala

ABSTRACT

It is always a very costly process to achieve a high level of reliability in software industry. So it is desirable for an industry to design software with minimum cost and optimal level of reliability. This paper addresses the problem of maximizing reliability of software by minimizing the overall cost of software. The technique deals with selecting various modules of application with optimal reliability of each to obtain desired level of reliability and cost of application. The problem is divided into three sections. Firstly it considered reliability of application as constraint to find optimal cost, secondly the cost is taken as constraint to find maximum amount of reliability for given cost constraint. At last the GA is used to find an optimal solution using both cost and reliability constraints exist.

KEYWORDS

RCR, DTMC, GA, development cost, complexity of the software.

1. INTRODUCTION

Software plays an important role in today's life, as software failure can lead to a huge economic loss or even endanger lives. Thus, reliability is a critical quality requirement of software and is an important measurement for use in quality assurance. Improving software reliability effectively requires that software design be refined and design defects be identified as early as possible in the software development life cycle so that cost and efforts can be reduced. Reliability is one of the essential quality requirements of software systems, especially for life critical ones. Software reliability modeling provides a means for estimating reliability of software, which facilitates effective decision making for quality control [1]. However, higher level reliability of software can definitely result in more development cost which, in turn, restricts the software distribution. It is concluded that a strong relationship exists between cost and reliability [2]. In practical engineering, software project managers are required to estimate the cost needed to complete the development of a software system taking into account the required level of reliability [3]. The area of the optimization of reliability allocation and development cost has gained more and more attention [4].

Various techniques had been used in the past for designing systems under dual and often conflicting constraints of maximizing reliability and minimizing cost. The idea of software reliability allocation was first put forward by M. E. Helander and NiclasOhlsson [5] they described a reliability allocation model called RCCM (Reliability Constrained Cost Minimization). Software reliability allocation plays an important role during software product design phase; this has close relationship with software modeling and cost evaluation. Architecture based approach have been used for the reliability assessment of software system [6-7]. This technique is also used for exploration of cost/reliability tradeoffs. It shows nonlinear but monotonic relation between the cost and reliability of individual components of that software. [8]

Indeed, the cost should be estimated, as early and accurately as possible, so that the development team can choose the proper techniques in order to address constraints of cost and required reliability. Several software reliability models had been proposed in the past decades to help software managers and developers to analyze and design systems under such dual and often conflicting constraints of maximizing reliability and minimizing cost [9-16]. In spite of there being many software reliability models, most of them focus on the cost estimation during testing phase of software development. The size and complexity of computer systems has increased more rapidly so to evaluate this various architectural approaches has been used. Techniques is used which characterize the behavior of such component based on their architecture [17]. Complex software systems are increasingly used in critical scenarios, such as medical devices, educational devices, neural network control. The complexity of such scenarios creates new challenges to software engineers who want to make system highly reliable while keeping cost low. Several techniques can be used to achieve the high reliability level while keeping the cost minimum. Reliability and cost evaluation as one part of software quality evaluation. Reliability is execution qualities. The reliability of module depends on its internal capabilities so there is a requirement for architecture evaluation of the module. The purpose of architecture evaluation is to identify potential failures and to verify that the quality requirements in design of software [18]. The architecture of a software application can be used to determine the reliability and cost of that application. In proposed work the software application architecture is represented with the help of DTMC, which is used to show the interdependency of modules with probability of visits

from one module to another. Then the cost and reliability of application using proposed genetic algorithm is calculated. Genetic algorithms are characterized by the natural process of evolution. They start with the initial population of solutions represented as chromosomes [19]. A chromosome comprises genes where each gene represents a specific attribute of the solution. The fitness of a chromosome is a measure of the quality of the solution it represents, in terms of various optimization parameters of the solution. In this paper genetic algorithm is used to provide optimal frame work and to evaluate cost and reliability tradeoffs. Choice of genetic algorithm motivated by three facts that is large and discontinuous search space, monotonic relation between the cost and reliability of individual component of that software [20].

2. ARCHITECTURE BASED SOFTWARE RELIABILITY AND COST ANALYSIS

A software system consists of n subsystems or modules within a given architecture and all the components are essential to the system and their failures are statistically independent. Therefore it is assumed that there exists a relationship between the total system reliability R and its components R_i (i=1, 2,…,n). it is also assumed that the reliability of a module is directly related to its cost. Various experiments have been conducted to show that an optimal or near optimal solution to the problem of selecting the component comprising the software can be obtained with lower cost and a high reliability of software, i.e. solution can be achieved in minimum cost. For this purpose the problem is divided in three parts as discussed in preceding sections.

2.1 Software Reliability Maximization Using Cost Constraint

Software reliability allocation plays an important role during software product design phase which has close relationship between reliability and cost. Software reliability is the probability that software will provide failure free operation in a fixed environment for a fixed interval of time. How to allocate reliability to each component so that system reliability can be maximized when cost is given. In this section the main objective is to achieve maximum reliability using cost as constraint. Considering the software reliability R and the expected software cost C as evaluation criteria the problem of software reliability maximization using cost as constraint is formulated as

$$\max_{R} f(R)$$

$$s.t.\, C \le C_0, 0.91 \le R_i \le 0.99$$

Where

$$R = \prod_{i=1}^{n} R_i^{v_i} \quad (1)$$

Indices:

i=module index, i=1, 2…n.

Parameters:

R_i = reliability of module i.

C=Overall cost of the software.

R=Overall Reliability of software.

Decision Parameters:

C_0=maximum cost allowable

The reliability of the application, denoted by R, during a single run in this case is given by equation1 [13].Where V_i is the number of times the application visits component i during a single run of the application. Since V, the number of visits to component i during a single run of the application is a random variable, the reliability of the application R in equation (1) is itself a random variable, and we are interested in its expected value denoted by E[R],which is given by equation2 [21].

$$E[R] = E[\prod_{i=1}^{n} R_i^{V_i}] = \prod_{i=1}^{n} E[R_i^{V_i}] \quad (2)$$

To find the value of E[R], there is a need to obtain $E[R_i^{V_i}]$ that is the expected reliability of component i during a single run of the application. The expected value of V denoted by V_i, and the variance of V_i denoted by σ_i^2, can be obtained from DTMC analysis.The effective expected reliability of component i, during a single run of the application is given by equation 3 [14].

$$E[R_i^{V_i}] = R_i^{v_i} + \frac{1}{2}(R_i^{v_i})(\log R_i)^2 \sigma_i^2 \quad (3)$$

Using equation (2) and (3) .the expected reliability of the application is given by:

$$E[R] = \prod_{i=1}^{n} R_i^{v_i} + \frac{1}{2}(R_i^{v_i})(\log R_i)^2 \sigma_i^2 \quad (4)$$

Equation4 implies that the expected reliability of the application depends not only on the reliabilities of its individual modules, but also on the expected number of times the application visits the module, and the variance of the number of visits.

2.2 Software Cost Minimization Using Reliability Constraint

In practical engineering, software project managers are required to estimate the cost needed to complete the development of a software system taking into account the required level of reliability. The area of the optimization of reliability allocation and development cost has gained more and more attention. Various techniques had been used in the past for designing systems under dual and often conflicting constraints of maximizing reliability and minimizing cost. In this section main objective is to minimize the cost taking reliability as a constraint. The objective function for given problem is formulated as given below.

$$\min_{C} f(C)$$

$s.t.\ R \geq R_0$

Where

$$C = \sum_{i=1}^{n} C_i \quad , \quad C_i = -\sum_{i=1}^{n} \frac{\alpha_i}{\ln r_i} + \beta \qquad (5)$$

Indices:

i=module index, i=1, 2…n.

Parameters:

C=overall cost of software.
R=overall reliability of software.
C_i = cost of module i.
α_i = compexity of the software.
β= development cost

Decision Parameters:

R_0=minimum reliability requirement of software.

Software development process consists of several phases, where estimation of development cost means different things with respect to different phases of software development life cycle. The cost of a module is associated with a number of parameters such as the design and development time, and testing time, and can be assumed to be monotonically related to the module reliability. The module cost-reliability relation can be linear, superliner, exponential, and even random while maintaining the monotonic property. The cost of the application is calculated using RCR model as given in equation 5. It can be very expensive to achieve the reliability value of 1. In some cases cost function derived from basic considerations and is usually stated in terms of the reliability.

2.3 Software Reliability and Cost Tradeoffs

In this section a set of optimal solutions is provided to the user by considering the constraints of reliability maximization using cost as constraint and cost minimization using reliability as constraint. The relationship between the cost and reliability of individual modules comprising the software is usually nonlinear but monotonic relation exists between them [22-23]. The optimization problem in this case is formulated as given below:

$$\min_{C} \max_{R} f(C, R)$$

$s.t.\ R \geq R_0$ and $C \leq C_0$

In this section an optimal solution satisfying both the constraints of cost and reliability is also found. The solution space may consist of a single solution, a set of solutions or no solution will be there for the given constraints. The problem is solved using genetic algorithm and the solutions are provided in preceding sections.

3. IMPLEMENTATION OF PROPOSED ALGORITHM

In this section the use of proposed genetic algorithm for obtaining solutions of above stated problems is explained. A problem from existing literature is selected to conduct the experiment. The application consists of 10 modules. It starts execution from first module and end at the execution of 10th module. The architecture of the problem is shown in figure1, and the inter-component transition probabilities are given in table 1 [24]. The visits statistics for the components of the application from DTMC analysis are presented in table 2. Matlab Genetic Algorithm toolbox is used in work to optimize objective functions. The various Genetic algorithm parameters used are given in table 3.

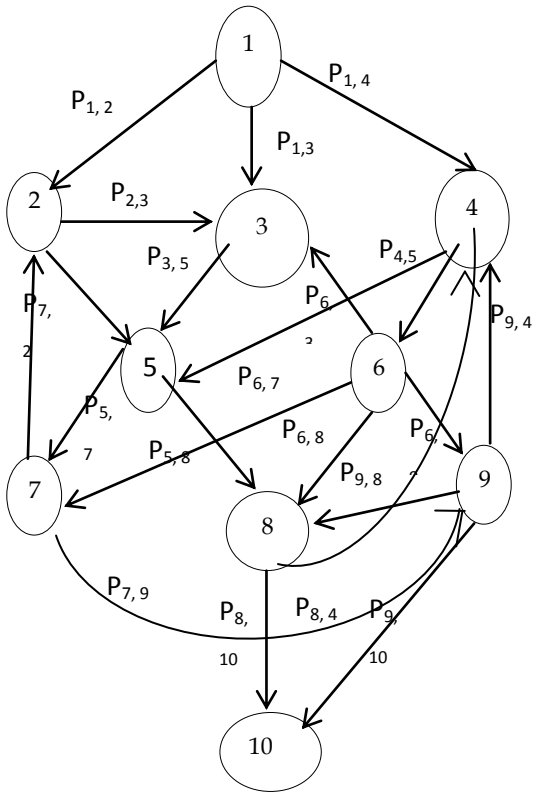

Figure1. Software Application with architecture described by absorbing DTMC

Table1. Inter-component transition probabilities

$P_{1,2}$=0.60	$P_{1,3}$=0.20	$P_{1,4}$=0.20	
$P_{2,3}$=0.70	$P_{2,5}$=0.30		
$P_{3,5}$=1.00			
$P_{4,5}$=0.40	$P_{4,6}$=0.60		
$P_{5,7}$=0.40	$P_{5,8}$=0.60		
$P_{6,3}$=0.30	$P_{6,7}$=0.30	$P_{6,8}$=0.10	$P_{6,9}$=0.30
$P_{7,2}$=0.50	$P_{7,9}$=0.50		
$P_{8,4}$=0.25	$P_{8,10}$=0.75		
$P_{9,8}$=0.10	$P_{9,10}$=0.90		

Table2. Visit statistics for the modules of the application.

Component #	Expected Number	Variance
1	1.0000	0.0000
2	0.9077	0.6444
3	0.9107	0.5499
4	0.4184	0.3928
5	1.3504	0.7185
6	0.2510	0.2319
7	0.6155	0.6261
8	0.8737	0.4225
9	0.3831	0.2462
10	1.0000	0.0000

Table3. Genetic Algorithm Parameters used in proposed study

Population type	Double vector
Population size	20
Scaling function	Rank
Selection function	Stochastic function
Reproduction: 1.Elite count	2
Crossover fraction	0.8
Crossover function	Scattered
Migration :1.Direction	Forward
2.Fraction	0.2
3.Interval	20
Evaluate	Fitness and constraint function in serial.

In the first study the attempt is to maximize the reliability of the software taking cost of the software as constraint. Here allowable cost (C_0) of the system isconsidered 951. The optimal solution obtained in this case is shown in table 4. From the results it is found that maximum reliability that can be achieved with the cost of 951 is 0.5651.

The last study focused on the issue when both the reliability and the cost of the application are taken as constraint. This is a multiobjective problem where R_0 is 0.82 and C_0 is 1030. There may be some situation where no solution will be available or more than one solution may be present, table6 shows a solution for the above defined constraints by taking the value of $\beta = 90$ and $\alpha = 0.3$.

Table4. An optimal design generated by proposed GA when C_0=951

COMPONENT	RELIABILITY	COST
1	0.9560	96.66
2	0.9120	93.25
3	0.9810	105.63
4	0.9310	94.19
5	0.9100	93.18
6	0.9100	93.18
7	0.9100	93.18
8	0.9380	94.68
9	0.9100	93.18
10	0.9100	93.18
	OVERALL:0.5	OVERALL:950.3
	651	1

In the second study algorithm is used to calculate the cost of application by taking the software reliability as constraint. Table5 shows the result of the problem when R_0 is considered 0.84.

Table5. An optimal design generated by proposed GA when R_0=0.84

COMPONENT	RELIABILITY	COST
1	0.9860	111.27
2	0.9740	101.38
3	0.9750	101.84
4	0.9860	111.27
5	0.9860	111.27
6	0.9420	95.02
7	0.9830	107.49
8	0.9800	104.84
9	0.9730	100.96
10	0.9770	102.89
	OVERALL:0.849	OVERALL:1048.23

Table6. An optimal design suggested by the proposed GA for Multiobjective problem.

COMPONENT	RELIABILITY	COST
1	0.9729	100.91
2	0.9587	97.11
3	0.9833	107.81
4	0.9704	99.98
5	0.9878	114.43
6	0.9510	95.97
7	0.9560	96.66
8	0.9734	101.12
9	0.9738	101.29
10	0.9863	111.74
	OVERALL:0.82019	OVERALL:1027.02

3. CONCLUSION AND FUTURE SCOPE

In this paper genetic algorithm is used to calculate optimal solution for the single objective and multiobjective problems of maximizing reliability of the software by minimizing the overall cost of the application. The sensitivity of the performance and reliability predictions of individual component are also analyzed in the study. In this paper reliability and cost equation is used to calculate overall reliability and overall cost respectively. Optimal solution is maximization of reliability taking cost as constraint and minimization of cost taking reliability as constraint. Optimal value of module is selected to achieve maximum reliability and minimum cost. In future it is intended to apply this technique to some realistic problems.

4. REFERENCES

[1] S.A Wadekar, S. G. 1991 Exploring Cost and Reliability Tradeoffs in Architectural Alternatives Using a Genetic Algorithm. In Proceedings of the 10th International Symposium on software Reliability Engineering, IEEE, 104-113 (1991).

[2] Wang W., W. Y. 1999 An Architecture-Based Software Reliability Model In Proc. of Pacific Rim International Symposium on Dependable Computing (1999).

[3] C.Y. Huang, M. 2005 Optimal Release Time For Software Systems Considering Cost, Testing-Effort, And Test Efficiency IEEE Transactions on Reliability, 54 (4), 583-59.

[4] C.Y.Huang, J. L. 2004 Optimal Allocation of Testing Resource Considering Cost, Reliability and Testing Effort In Proceedings of Pacific Rim Dependable Computing, 103-112 (2004).

[5] Guan, H. 2009 Exploring Architecture-Based Software Reliability Allocation Using ADynamic Programming Algorithm. International Computer Science and Computational Technology, 106-109,

[6] M.E Helander, M. 1998 Planning Models for Software Reliability and Cost. IEEE Trans. on Software Engineering.

[7] Wang. W L, P. D. 2006 Architecture-Based Software Reliability Modeling. The Journal of System and Software (79), 132-146.

[8] Chin-Yu Huang, J.-H. L.-Y. 1999 Software Reliability Modeling and Cost EstimationIncorporating Testing-Effort and Efficiency In Proceedings of 10th international symposium on Software Reliability Engineering, 62-72 (1999).

[9] Goldberg, D. Genetic Algorithms in Search, Optimization and Machine Learning,Addison-Wesley Pub.Co. (1989).

[10] Hui Guan, W.-R. C.-J. 2010 Estimation of Reliability and Cost Relationship for Architecture Based Software. International Journal of Automation and Computing.

[11] Marko Palviainen, A. E. 2011. The Reliability Estimation, Prediction and Measuring of Component-Based Software. The Journal of Systems and Software (84), 1054-1070.

[12] Roberto Pietrantuono, S. R. 2010 Software Reliability and Testing Time Allocation: An Architecture-Based Approach. IEEE Rransaction on Software Engineering, 36 (3).

[13] S. Gokhale, M. R. 1998 Reliability Simulation of Component-Based Software Systems. In Proc. of 9th Intel. Symposium on Software Reliability Engineering, 192-201 (1998).

[14] S.Kirkpatrick, C. G. 1983 Optimization by Simulated Annealing. Science (220), 671-680.

[15] S.Y.Kuo, C. L. 2001 A Framework for Modeling Software Reliability using Various Testing Efforts and Fault Detection Rates. IEEE Transactions on Reliability, 50, 310-320.

[16] W.Ning, Y. A. 2002Balance between Software Reliability and Cost. Systems Engineering and Electronics, 24, 117-119.

[17] F.Zahedi, N. 1991 Software Reliability Allocation based on Structure, Utility, Price and Cost. IEEE Trans on Software Engineering, 17(4), 345-355.

[18] C.Y. Huang, J. L. 1999. Software Reliability Modeling and Cost Estimation Incorporating Testing-Effort and Efficiency. In Proceedings of the 10th International Symposium on Software Reliability Engineering, 62-72 (1999).

[19] D.W Coit, A. S. 1996 Reliability Optimization of Series-Parallel Systems Using a Genetic Algorithm. IEEE Trans. on Reliability, 45(2), 254-266.

[20] L.Painton, J. 1995 Genetic Algorithms in Optimization of System Reliability. IEEE Trans. on Reliability, 44(2), 172-178.

[21] S.Gokhale, K. T. 2002. Reliability Prediction and Sensitivity Analysis based on Software Architecture. In Proc. Of Intl. Symposium on Software Reliability Engineering (ISSRE 02) (Nov, 2002).

[22] S.Gokhale, S. 2004 Cost Constrained Reliability Maximization of Software Systems. IEEE Annual Symposium-Rams on Reliability and Maintainability, 26-29 (Jan, 2004).

[23] Luenberger, D. G.2003. Introduction to Linear and Nonlinear Programming. Kluwer Academics Publishing Group, 2nd Edition

[24] R.Salceao, M. G. 1990. An Improved Random-Search Algorithm for Non-Linear Optimization. Computers and Chemical Engineering, (14), 1111-1126.