ISBN 978-952-5726-07-7 (Print), 978-952-5726-08-4 (CD-ROM)
Proceedings of the Second Symposium International Computer Science and Computational Technology(ISCSCT '09)
Huangshan, P. R. China, 26-28,Dec. 2009, pp. 106-109

Exploring Architecture-Based Software Reliability Allocation Using a Dynamic Programming Algorithm

Hui Guan[1,2], Tingmei Wang[3], and Weiru Chen[1]
[1] Shenyang Institute of Chemical Technology, Shenyang, China
[2] STRL, De Montfort University, Leicester, England
Email: guanh1999@126.com
[3] Beijing Union University, Beijing, China
Email:wtm9329@hotmail.com

***Abstract*—Software reliability allocation plays an important role during software product design phase, which has close relationship with software modeling and cost evaluation. We formulated an architecture-based approach for modeling software reliability optimization problem, on this basis a dynamic programming algorithm has been illustrated in this paper which can be used to allocate the reliability to each component so as to minimize the cost of designing software while meeting the desired reliability goal. The result of our experiment show an optimal or near optimal solution to the problem of selecting the component comprising the software can be obtained with lower cost..**

***Index Terms*—Architecture, Software Reliability, Reliability Allocation, Dynamic Programming**

I. INTRODUCTION

The impact of software structure on its reliability and correctness was highlighted as early as 1975-76 [1, 2]. However, with the rapid expansion of software system size and complexity, the software structure has shifted from the earlier structure- oriented systems to object-oriented one of the present and of the future, which has also triggered a number of efforts in development of various techniques for evaluation of these systems.

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. The idea of software reliability allocation was first put forward by M.E. HELANDER and Niclas Ohlsson [3], they described a reliability allocation model called RCCM (Reliability Constrained Cost Minimization), which is used to assign the reliability. In addition, Zahedi and Ashrafi [4] adopt AHP method for modeling the software architecture with cost as the constraints and propose a model relating to the system reliability maximization. Boehm [5] presents a method for evaluating software development cost by using COCOMO model.[6, 7] outline a method for optimization of reliability allocation and testing schedule for a software system taking into account the reliability growth of its components.

However, the models and methods presented above are mainly applied during the late phase of software development other than the early stage. It is known that the later the defects are found, the higher cost needs to be paid for them. Software architecture is just the product of early stage in software development. If the development cost evaluation can be applied in accordance with the given software architecture and reliability requirement and be implemented before the software development, development resources can be assigned properly and well organized software development can be guaranteed under such a circumstance. The aim of this paper is to propose an idea of architecture-based software reliability allocation, address an architecture-based software reliability-cost model for evaluating the architecture before developing the software.

The discussion of this paper takes the following structure. Section 2 introduces architecture-based software reliability allocation model. Section 3 depicts how to find out the optimal allocation method by using a dynamic programming algorithm. Section 4 illustrates the application of the algorithm proposed in section 3 and section 5 offers concluding remarks and directions for future research.

II. SOFTWARE RELIABILITY ALLOCATION MODEL

A. Software development cost minimization versus reliability allocation

In fact, it is impossible to improve the software reliability while lowering the software system development cost because they are two conflicting constraints.

The so-called software development cost minimization can be considered from two points of views. One is to find an optimal reliability allocation method while achieving the given reliability such that the development cost can be as low as possible; the other is how to allocate the reliability to each component on the premise of the given cost so that the system reliability can be maximized. This paper focuses on the former one.

Many systems are implemented by using a set of interconnected subsystems. Reliability allocation means adjusting the reliability among different subsystems so that the total system development cost (including human, material resources, development time and testing time etc) can be minimized. Reliability allocation can be used to

deal with such kind of problem that the goal is set prior to the solution. Usually the number of the solution is more than one, as a result reliability allocation is used to deal with the optimal problem with some constraints.

B. Software reliability and cost model

It is reasonable to assume that the cost function f_i would satisfy these three conditions [8]:

- f_i is a positive definite function
- f_i is non-decreasing
- f_i increases at a higher rate for higher values of R_i

The third condition suggests that it can be very expensive to achieve the reliability value of 1. In fact for software, it has been shown that under some assumptions, it is infeasible to achieve ultra-high reliability in software [9].

In some cases, the cost function can be derived from basic considerations and is usually stated in terms of the reliability, as we will do below for software reliability.

According to the definition of software reliability [10]:

$$r(t) = e^{-\lambda t} \qquad (1)$$

Where r(t) is continuous-time system reliability, and λ is its failure rate.

From the formula above, it can be concluded that the reliability of a software is strongly dependent on the number of faults that remain in it after testing and debugging have finished; as fault decreases the probability that the software works according to its specification will increase, that is, reliability will vary inversely with faults.

According to the experiences from practical engineering, this paper takes account of the factors by assuming that software reliability and development cost satisfies the relations as below:

1) The number of faults existed in system is inversely proportional to system development cost, and the system failure rate is proportional to the number of faults. The relationship between them can be presented as:

$$E \propto 1/C \qquad (2)$$

$$\lambda \propto E \qquad (3)$$

Where C is the development cost and E is the number of system failures. Using equation (2), reliability function (1) is therefore

$$C \propto \frac{-1}{\ln r} \qquad (4)$$

2) Software development cost is proportional to the complexity and size of the software system.

Taking assumption 2) into account, the equation (4) can be expressed as:

$$C = \frac{-\alpha}{\ln r} \qquad (5)$$

Where α represents the complexity and size for developing the software system and is called reliability-cost coefficient in this paper.

In consideration of the operation in practical engineering, where usually exists some basic cost such as personnel training, development tools preparation etc. We use character β to represent the basic development cost in this paper. With the assumption above, we describe the reliability-cost model as below:

$$C(r) = \frac{-\alpha}{\ln r} + \beta \qquad (6)$$

Equation (6) states the relationship between the system development cost and the system reliability in our model. While determining the parameters α and β are more time-consuming and challenging, however, they can be obtained based on the prior experience of practical engineering. With regard to the basic indivisible software or component and relative to a specified software development group, α and β are the fixed values.

Software reliability-cost model is shown in the Fig.1, where r and C refer to the software reliability and the development cost. According to the model, it concluded that it will cost much in order to achieve a very high level reliability. From the following figure we can find out that the software with 100% reliability is impossible.

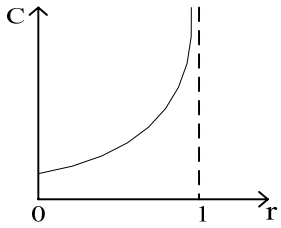

Figure 1. Reliability-cost model

Given a system with many components, the reliability of its component$_i$ can also be stated as:

$$C(r_i) = \frac{-\alpha_i}{\ln r_i} + \beta_i \qquad (7)$$

C. Software reliability allocation model

Here we assume the software system has been designed as an assembly of appropriated connected components. Let there be n components, each with reliability r_i and cost C_i, i=1…n.. Let R_{obj} be the specified target reliability and r and C represent the overall reliability and the total system cost. Let $F(r_1,r_2,\ldots r_m)$ be the function of r and r_i. The software reliability allocation model can be stated as:

Objective function

$$\min C \tag{8}$$

Subject to

$$\begin{cases} r = F(r_1, r_{2,}, ..., r_n) \geq R_{obj} \\ 0 < r_i < 1 \end{cases} \tag{9}$$

$$C = \sum C_i = \sum \beta_i - \sum \frac{\alpha_i}{\ln r_i} \tag{10}$$

The case we have discussed above is a nonlinear combination optimization problem with one object function and more than one constraint set. That is to say: try to reallocate the reliability of each subsystem or components so as to achieve the target reliability while minimizing the total development cost. Because the cost base value β of the components is nothing to do with reliability and has no impact on achieving the minimal cost, β is simplified to 0. The constraint in equation (10) can be rewriting as:

$$C = \sum C_i = -\sum \frac{\alpha_i}{\ln r_i} \tag{11}$$

As for a software system with n subsystems or components within a given architecture, the development cost of the system will increase while enhancing the software reliability target. Significantly, the greater improvement of the reliability, the more increase of the cost will be required. As shown in figure 1, the relationship between them is nonlinear.

Such a problem can be resolved with variety methods, and dynamic programming is an option.

III. USING DYNAMIC PROGRAMMING ALGORITHM TO SOLVE RELIABILITY ALLOCATION PROBLEM

Given a software system with n components and the relationship function F discussed above is known. The reliability-cost coefficient α of each component and the specified system reliability target R_{obj} is given.

The dynamic programming algorithm is as follows:

Step 1: Let S represent the reliability matrix $[r_1, r_2, ., r_n]$, T represent the cost matrix $[c_1, c_2, \ldots, c_n]$, δ be the solving step length, I_i represent the matrix with one column and n rows in which only the value of the ith element is 1 and the rest are all 0. Assume S_0=[maxr, maxr, …, maxr], maxr represents the maximized possible reliability, for example 0.9999, which means the initial reliability values of the components are all maxr.

Step 2: As for S_0, T_0 can be solved by (11), C_0 can be given by (11) and system reliability R_0 can be given by function F.

Step 3: If $R_0 < R_{obj}$ then stop and return. No solutions.

Step 4: Set Rate=0;

Step 5: for i=1 to n

i) $S' = S_0 - I_i * \delta$;

ii) With regard to S', Generate reliability R' with the function F, T' with (7), total cost C' with (11)

iii) $\Delta C = C_0 - C'$; $\Delta R = R_0 - R'$;

iv) if $R' \geq R_{obj}$ and $\Delta C/\Delta R >$Rate then Set Rate=$\Delta C/\Delta R$, R=R' , S=S' , C=C' , T=T';

Step 6: if $R_0 \neq R$ then set S_0=S; R_0=R; C_0=C; T_0=T; return to step 4

Where reliability allocation result S_0 is the reliability of each component. R_0 and C_0 are the corresponding system reliability and expected system development cost. T_0 is the expected development cost allocated to each component.

Notice from the above that prerequisite to the correctness of the algorithm is that the decrease in reliability of one component can result in that of the whole system and lower the development cost. But that can be guaranteed in our algorithm. The aim of step 5 iv) in the above algorithm is to select an optimal component whose decrease in reliability can result in the maximal cost/reliability variation, which makes the single step programming optimized so that optimal reliability allocation of the ultimate system is guaranteed.

IV. EXAMPLE AND RESULT

Here we choose a system with three independent components r_1, r_2, r_3. We assume that all the components are essential to the system and their failures are statistically independent. Therefore, the relationship between the total system reliability r and its components' reliability r_i (i=1, 2, 3) can be stated as: $r = F(r_1, r_2, r_3) = r_1 * r_2 * r_3$ Suppose that the complexities of the components are 0.30, 0.72 and 0.54 respectively. In order to minimize the system development cost and the system reliability shall be no less than 0.95, how to allocate the reliability to each component. Set the precision of computing is 0.01.

Such a problem can be rewritten as:

$R = r_1 * r_2 * r_3 \leq 0.95$

$c_1 = -0.30/\ln r_1$

$c_2 = -0.72/\ln r_2$

$c_3 = -0.54/\ln r_3$

Compute the values of parameters (r_1, r_2, r_3) with which the total cost C ($C = c_1 + c_2 + c_3$) is minimized.

With respect to each component, we compute the cost with the reliability from 0.95 to 0.99 (increment is 0.01) according to the reliability/cost function model in the data set as shown in Table 1.

TABLE I. COST AND RELIABILITY DATA SET

	r_1	c_1	r_2	c_2	r_3	c_3
1	0.95	5.85	0.95	14.04	0.95	10.53
2	0.96	7.35	0.96	17.64	0.96	13.23
3	0.97	9.85	0.97	23.64	0.97	17.73
4	0.98	14.85	0.98	35.64	0.98	26.73
5	0.99	29.85	0.99	71.64	0.99	53.73

According to the algorithm above, set initial state S_0 = [0.99, 0.99, 0.99]. Accordingly, T_0 = [29.85, 71.64,

53.73], δ=0.01, and the system cost C_0= 29.85 + 71.64 + 53.73 = 155.22, system reliability R_0= 0.99 * 0.99*0.99 =0.97.

Set i=1, 2, 3, then compute separately with different value:

1) S'= S_0 – [0.01, 0, 0] = [0.98, 0.99, 0.99], R'=0.96, T'= [14.85, 71.64, 53.73], C'=140.22

ΔC= 15, ΔR=0.01, ΔC/ΔR=1500

2) S'= S_0 – [0, 0.01, 0] = [0.99, 0.98, 0.99], R'=0.96, T'= [29.85, 35.64, 53.73], C'=119.22

ΔC= 36, ΔR=0.01, ΔC/ΔR=3600

3) S'= S_0 – [0, 0, 0.01] = [0.99, 0.99, 0.98], R'=0.96, T'= [29.85, 71.64, 26.73], C'=128.22

ΔC= 27; ΔR=0.01, ΔC/ΔR=2700

Choose the optimal result 2), set S_0= [0.99, 0.98, 0.99], continue to perform the same operation :

1) S'= S_0 – [0.01, 0, 0] = [0.98, 0.98, 0.99], R'=0.95, T'= [14.85, 35.64, 53.73], C'=104.22

ΔC= 15, ΔR=0.01, ΔC/ΔR=1500

2) S'= S_0 – [0, 0.01, 0] = [0.99, 0.97, 0.99], R'=0.95, T'= [29.85, 23.64, 53.73], C'=107.22

ΔC= 12, ΔR=0.01, ΔC/ΔR=1200

3) S'= S_0 – [0, 0, 0.01] = [0.99, 0.98, 0.98], R'=0.95, T'= [29.85, 35.64, 26.73], C'=92.22

ΔC= 27; ΔR=0.01, ΔC/ΔR=2700

Choose the optimal result 3), set S_0= [0.99, 0.98, 0.98], and then continue to perform the same operation, all of the results R' are less than the specified reliability target 0.95. Therefore, the reliability allocation in this case is as below:

1) System reliability allocation S_0= [0.99, 0.98, 0.98];

2) System reliability R_0=0.95;

3) Expected system development cost C_0= 92.22;

4) Expected development cost assigned to each components T_0 = [29.85, 35.64, 26.73].

V. CONCLUSION AND FUTURE WORK

Software reliability allocation plays an important role during software product design, which has close relationship with software modeling and cost evaluation [11]. We formulated an architecture-based approach for modeling software reliability optimization problem, on this basis a dynamic programming algorithm has been illustrated in this paper which can be used to allocate the reliability to each component so as to minimize the cost of designing software while meeting the desired reliability goal. The result of our experiment show an optimal or approximate optimal solution to the problem of selecting the component comprising a software can be obtained with lower cost (a high reliability). The reliability and cost allocation model presented in this paper can be used to solve the optimal allocation problems in simple systems, it is also applicable in complex systems.

We did not consider how to set the value of solving step length δ in this paper. However, the greater of δ will result in the imprecise solving result and the smaller of δ will slow down the speed of solving. We can adopt a technique of variable step length relating to solving precision to resolve such problem. Another problem is to do with setting the initial value for S_0. In our algorithm, we suppose a reliability value *maxr* with maximal possibility. But how much should the value be? Whether should an initial value be set in favor of solving more quickly? These problems deserves further studied.

REFERENCES

[1] D.L.Parnas. "The Influence of Software Structure on Reliability". In *Proc.1975 Int'l Conf. Reliability software*, Los Angeles, CA, April 1975. pp. 358-362.

[2] M.L.Shooman. "Structural models for software reliability prediction". In *Proc. 2nd Int'l Conf. Software Engineering,* San Fransisco, CA, October 1976, pp. 268-280.

[3] M.E. HELANDER, M. Zhao and N. Ohlsson. "Planning Models for Software Reliability and Cost". *IEEE Trans. on Software Engineering,* 1998, 24(6):420~434.

[4] F. Zahedi and N. Ashrafi, "Software Reliability Allocation Based on Structure , Utility , Price and Cost ". *IEEE Trans. on Software Engineering,* 1991, 17 (4):345 - 356.

[5] B. Boehm , R. Valerdi , J A. Lane et al, *COCOMO Suite Methodology and Evolution*, CrossTalk, 2005, pp. 20 - 25.

[6] C. Y. Huang, J. H. Lo and S Y. Kuo, "Optimal Allocation of Testing resource Considering Cost, Reliability, and Testing Effort", In *Prof. 2004 Pacific Rim Dependable Computing*, French Polynesia, 2004, pp.103 - 112.

[7] S. Y. Kuo, C. Y. Huang and M R. Lyu, "A Framework for Modeling Software Reliability , Using Various Testing Efforts and Fault Detection Rates". *IEEE Transactions on Reliability*, 2001, 50(3) :310 - 320.

[8] A. Mettas, Reliability allocation and optimization for complex systems. In *Proc. Annual Reliability and Maintainability Symposium,* Los Angeles, CA, January 2000,pp.216-221.

[9] R. W. Bulter and G.B. Finelli, "The infeasibility of quantifying the reliability of life-critical real-time software", *IEEE Trans. on Software Engineering*, 1993,19:3-12.

[10] M. R. Lyu. *Handbook of Software Reliability Engineering*. IEEE Computer Society Press, New York, l996, pp.36.

[11] M. R. Lyu. *Handbook of Software Reliability Engineering*. IEEE Computer Society Press, New York, l996, pp.3l5