BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **14**, No 3

Sofia • 2014 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.2478/cait-2014-0028

Software Reliability Estimate with Duplicated Components Based on Connection Structure

Zhen Li, Junfeng Tian, Pengyuan Zhao

College of Mathematics and Computer, Hebei University, Baoding 071002, Hebei, China
Emails: lizhen_hbu@126.com tjf@hbu.edu.cn zhaopengyuan@cmc.hbu.cn

Abstract: *Reliability testing of complex software at the system level is impossible due to the environmental constraint or the time limitation, so its reliability estimate is often obtained based on the reliability of subsystems or components. The connection structure was defined and the component-based software reliability was estimated based on it. For the present popular software with duplicated components, an approach to variance estimation of software reliability for complex structure systems was proposed, which has improved the hierarchical decomposition approach of variance estimation just for series-parallel systems. Experimental results indicated that the approach to variance estimation for reliability of software with duplicated components has advantages, such as the simple calculation process, small error result, and suitability for complex structure systems. Finally, the sensitivity analysis, used to identify critical components for resource allocation, could better improve the software reliability.*

Keywords: *Component-based software, reliability, duplicated components, variance estimation, complex structure.*

1. Introduction

For complex software, reliability testing at the system level is impossible due to the environmental constraint or the time limitation. In these situations, reliability can be estimated by conducting testing at subsystem or component levels. The software reliability estimate possesses uncertainty. In practice, people usually would rather select the software with lower and more accuracy reliability estimate than software with higher and less accuracy reliability estimate.

A common practice in software design is to use the same components in different locations of the software due to the requirement of the same or similar functions. For the same components, their reliability is often inferred from the same testing sample, that is, the reliability estimates of the same component are *s*-dependent. For software with duplicated components, the variance of the software reliability estimate needs computation of higher order moments involving iterations and discrete convolutions. These procedures are often time-consuming. If we ignore the dependence of the component reliability estimates, the variance of the software reliability estimate will be underestimated [1, 2].

In recent years, researches on component-based software reliability have been paid more and more attention to. M o h a m e d, Z u l k e r n i n e [3] have proposed a simple CFG structure that represents inter-component and intra-component control flow transitions; W a n g et al. [4] proposed a software reliability model which can deal with the cases of component interaction; F i o n d e l l a et al. [5] presented an efficient, scalable approach to analyze the reliability of a component-based software system considering the correlated component failures. These researches can estimate the component-based software reliability, but do not consider how to estimate its uncertainty.

For the uncertainty measurement of software reliability estimates, C o i t [6] demonstrated a flexible procedure to determine the confidence intervals for series-parallel system reliability, when there was uncertainty regarding the component reliability information; Jin expanded Coit's method to estimate the confidence intervals for series-parallel systems with arbitrarily repeated components [1], and proposed a hierarchical decomposition procedure to determine the variance of the reliability estimate for series-parallel systems [2]. H i l l et al. [7] presented probability inequalities and results useful in defining the inequality-based reliability estimate for series-parallel system with repeated components. The above approaches are just for series-parallel systems and inadequate for complex structure systems.

On the basis of the above problems, the connection structure is defined and the component-based software reliability is estimated based on it. For software with duplicated components, an approach to variance estimation of software reliability for complex structure systems is proposed, which improves the hierarchical decomposition approach of variance estimation just for series-parallel systems proposed by J i n [2].

2. A connection structure for component-based software

First of all, the component-based software is hierarchized to multi layers and each layer involves several modules, that is, the software is devised into multiple modules in a hierarchical order from the system level down to the component level. The rule to define a module is in two parts: 1) at least one component is included; and 2) within a module, there is only one connection structure for sub-modules or components. The connection structure is defined as follows:

Definition 1. A connection structure A is defined as the connection structure of modules or components in the same layer for the hierarchical component-based software, denoted by A=⟨S, P, F, B, L, SC, BC, LC⟩ where the elements represent a Sequence structure, Parallel structure, Fault tolerance structure, Branch structure, Loop structure, Sequence Call structure, Branch Call structure and Loop Call structure respectively. If a module contains only one component, the default connection structure is S, assuming that the component connects to itself.

Definition 2. A connection structure flowchart is the running flow chart of the hierarchical component-based software which expresses the connection structure of modules or components in the same layer.

Definition 3. The component-based software CS is denoted by $\mathrm{CS} = \{C(i), M(i), A(i) \mid i = 1, 2, ..., m\}$ where *m* is the number of layers, $C(i)$ is the set of the *i*-th layer components, $M(i)$ is the set of the *i*-th layer modules, $A(i)$ is the set of connection structures of the *i*-th layer modules or components.

We assume that the transfer of control among components is a Markov process. Let the reliability of components c_i $i=1, 2, ..., n$, be r_i and the transition probability from c_i to c_j be $p_{i,j}$, $i, j=1, 2, ..., n$, $p_{i,j} \in [0, 1]$.

2.1. Sequence structure

Components of the sequence structure are executed in sequence. The component sequence reliability is the product of component reliability. The sequence's probability of occurrence is the product of each transition probability. Then the reliability of the sequence structure can be expressed as the product of component sequence reliability and the sequence's probability of occurrence. In Fig. 1(a), the reliability of a sequence structure R_S is

$$R_\mathrm{S} = p_{1,2}p_{2,3}...p_{n-1,n}r_1r_2...r_n = \prod_{i=1}^{n-1} p_{i,i+1} \times \prod_{j=1}^{n} r_j. \tag{1}$$

2.2. Parallel structure

In concurrent environment, the performance of the system can be improved by running multiple components concurrently. The reliability of a parallel structure can be expressed as the product of component reliability. In Fig. 1b, the reliability of the parallel structure R_P is

$$R_\mathrm{P} = r_1r_2...r_n = \prod_{i=1}^{n} r_i. \tag{2}$$

2.3. Fault tolerance structure

Only one of the components with a fault tolerance structure runs at a certain time. In Fig. 1c, c_1 c_2,…, c_n , denoted by a dashed line, are the backup components of the primary component c_1. When all components in Fig. 1c fail, the fault tolerance structure is unreliable. The reliability of fault tolerance structure showed in Fig. 1c R_F is

$$R_\mathrm{F} = 1 - (1-r_1)(1-r_2)...(1-r_n) = 1 - \prod_{i=1}^{n} (1-r_i). \tag{3}$$

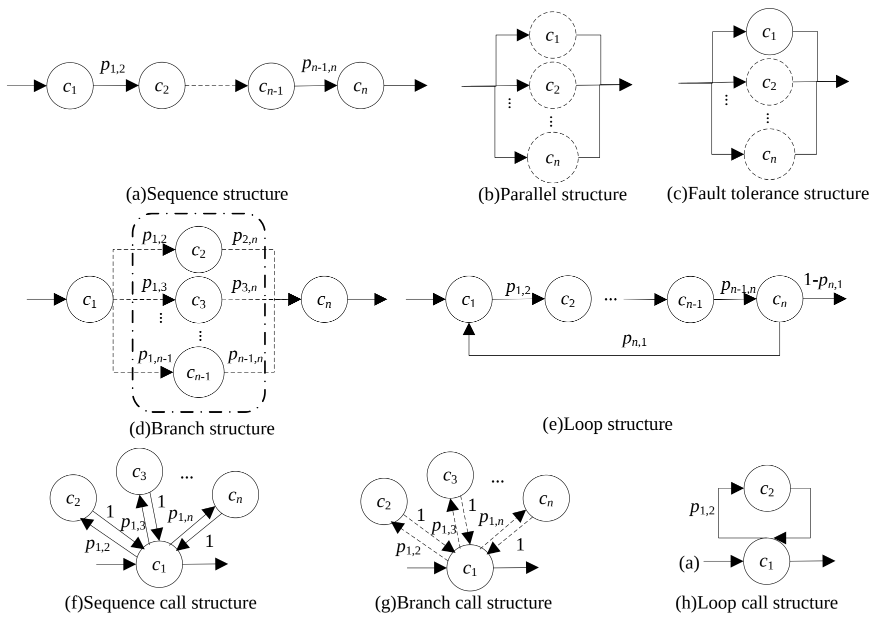


Fig. 1. A connection structure

2.4. Branch structure

In the branch structure, only one branch runs at a certain time. In Fig. 1d, the branch structure is denoted by a dash-dotted line. The reliability of the branch structure can be expressed as the sum of the product of branch reliability and the branch probability of occurrence. The reliability of the branch structure R_B is

(4) $$R_B = p_{1,2}p_{2,n}r_2 + p_{1,3}p_{3,n}r_3 + ... + p_{1,n-1}p_{n-1,n}r_{n-1} = \sum_{i=2}^{n-1} p_{1,i}p_{i,n}r_i.$$

2.5. Loop structure

Components with a loop structure are executed circularly. We add an entry component before the loop structure and an exit component after the loop structure in Fig. 1e. The reliability of the entry component r_{entry} and the exit component r_{exit} is 1. Then the reliable state *R* and the unreliable state *U* are added as terminal states, representing the state of the reliable and unreliable output respectively, shown in Fig. 2; $p_{i,j}r_i$ represent the probability that the execution of c_i produces the correct result and the control is transferred to c_j. We can gain the transition probability matrix $\boldsymbol{Q}$; $\boldsymbol{Q}^k(1, n+2)$, the element of row 1, column *n*+2 for the *k*-th power of matrix $\boldsymbol{Q}$ represents the probability that starting from an entry component, the chain enters the absorbing state {*R*, *U*} at or before the *k*-th step. The value range of *k* is $[0, \infty)$. If there is at least one component c_1 then $\boldsymbol{Q}^0(1, n+2) = \boldsymbol{Q}^1(1, n+2) = 0$. Let $\boldsymbol{M} = \sum_{k=0}^{\infty} \boldsymbol{Q}^k = (\boldsymbol{I} - \boldsymbol{Q})^{-1}$, $\boldsymbol{M}(1, n+2)$, the element of row 1, column *n*+2 for matrix

M, is the probability from an entry to an exit component. The reliability of the loop structure in Fig. 1e R_L is

$$R_\mathrm{L} = \boldsymbol{M}(1, n+2)r_\mathrm{exit} =$$

(5)
$$\begin{cases} \left[(1-p_{1,1})r_1\right] \Big/ \left[1-p_{1,1}r_1\right], & n=1; \\ \left[(1-p_{n,1}) \times \prod_{i=1}^{n-1} p_{i,i+1} \times \prod_{j=1}^{n} r_j\right] \Big/ \left[1-p_{n,1} \times \prod_{i=1}^{n-1} p_{i,i+1} \times \prod_{j=1}^{n} r_j\right], & n>1. \end{cases}$$

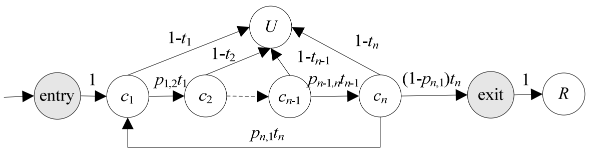


Fig. 2. A loop structure after adding an entry, exit and states

2.6. Call structure

There are dependent relations among the components with a call structure. We adopt the Continuation Passing Style (CPS) to transform these dependent relations. CPS transformation makes the caller component divided into several program segments, as sub-components, each of them does not calling other components. There are three types of call structures.

Type 1. Sequence call structure. The dependency relation in a sequence call structure is a sequence. In Fig. 1f, c_1 calls $c_2, \ldots, c_n$ in the sequence and the transition probability from c_2, or c_3,…,or c_n to c_1 is 1. c_1 is divided into sub-components $c_{1,1}, c_{1,2}, \ldots, c_{1,n}$. The input of c_2 is the output of $c_{1,1}$, the input of $c_{1,2}$ is the output of c_2, …, as shown in Fig. 3a. According to (1), the reliability of the sequence call structure R_SC is

(6)
$$R_\mathrm{SC} = p_{1,2}p_{1,3}...p_{1,n}r_{1,1}r_{1,2}...r_{1,n}r_2...r_n = \prod_{i=2}^{n} p_{1,i} \times \prod_{j=1}^{n} r_{1,j} \times \prod_{k=2}^{n} r_k, ,$$

where $r_{1,1}, r_{1,2}, \ldots, r_{1,n}$ is the reliability of $c_{1,1}, c_{1,2}, \ldots, c_{1,n}$ respectively.

Type 2. Branch call structure. The dependency relation in a branch call structure is a branch. In Fig. 1g c_1 calls c_2, or c_3, …, or c_n and the transition probability from $c_2, c_3, \ldots, c_n$ to c_1 is 1. c_1 is divided into $c_{1,1}$ and $c_{1,2}$. In Fig. 3b, $c_{1,1}$ is the part of c_1 before the branch and $c_{1,2}$ is the part of c_1 after the branch. The reliability of the branch structure can be calculated by (4). The reliability of the branch call structure R_BC is

(7)
$$R_\mathrm{BC} = r_{1,1}r_{1,2}(p_{1,2}r_2 + p_{1,3}r_3 + \ldots + p_{1,n}r_n) = r_{1,1}r_{1,2}\sum_{i=2}^{n} p_{1,i}r_i.$$

Type 3. Loop call structure. The dependency relation in a loop call structure is a loop. In Fig. 1h, c_1 calls c_2 circularly and the transition probability from c_2 to

c_1 is 1. c_1 is divided into $c_{1,1}$ and $c_{1,2}$. In Fig. 3c, $c_{1,1}$ is the part of c_1 before calling c_2 circularly, and $c_{1,2}$ is the part of c_1 after calling c_2 circularly. The reliability of the loop structure can be calculated by (5). The reliability of the loop call structure R_{LC} is

(8) $R_{\mathrm{LC}} = r_{1,1}r_{1,2}p_{1,2} \times \left[(1-p_{1,2})r_2\right] / \left(1-p_{2,1}r_2\right) = \left[p_{1,2}(1-p_{1,2})r_{1,1}r_{1,2}r_2\right] / \left(1-p_{2,1}r_2\right).$

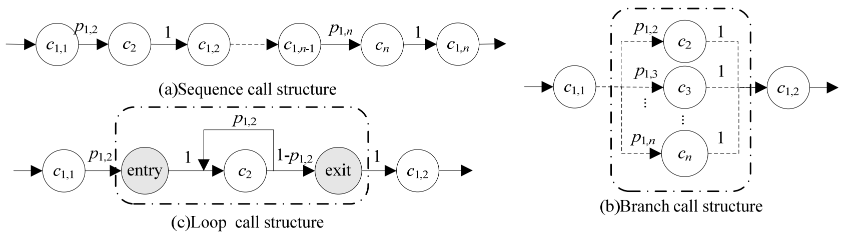


Fig. 3. A call structure after division

3. Reliability estimate of component-based software

It is easily proved that any component-based software can be expressed by a connection structure flowchart. Without loss of generality, the connection structure flowchart of component-based software in Fig. 4a is used as an example. The main modules are denoted by a dash-dotted line in each layer. The layer 0 of software is denoted by S_0; $S_{u,v}$ $u, v = 1, 2, \ldots,$ indicating that the sub-modules or components are connected in a sequence at its lower adjacent layer and it is the *v*-th module in layer *u*. The first subscript indicates the layer number, while the second subscript indicates the module number on that layer. Modules with other connection structures can be indicated by a similar expression. The hierarchical model of component-based software is shown in Fig. 4b.

To standardize the hierarchy, artificial modules, denoted by a dashed line, are introduced to the intermediate layers, such that all modules at one layer can only communicate with their adjacent layers. The overall software reliability *R* in Fig. 4a can be expressed recursively as

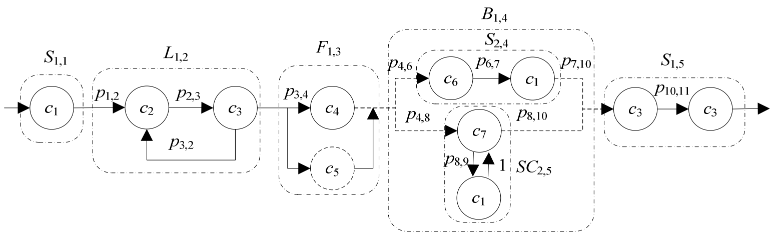
(9)
$$\begin{gathered} R = R_{S_0}(R_{S_{1,1}}[R_{S_{2,1}}(r_1)], R_{L_{1,2}}[R_{L_{2,2}}(r_2, r_3)], R_{F_{1,3}}[R_{F_{2,3}}(r_4, r_5)], \\ R_{B_{1,4}}[R_{S_{2,4}}(r_1, r_6), R_{SC_{2,5}}(r_7, r_1)], R_{S_{1,5}}[R_{S_{2,6}}(r_3)]). \end{gathered}$$

The true component reliability r_i, $i = 1, 2, \ldots, 7$, is usually not available, and the estimate $\hat{r}_i$ is often used to substitute r_i. For c_i, *n* samples are tested for a period of time *t* and the number of failures is *f*. The reliability of c_i can be estimated using a binomial distribution [6]: $\hat{r}_i = (n-f)/n$, $\hat{\mathrm{var}}(\hat{r}_i) = \left[\hat{r}_i(1-\hat{r}_i)\right]/n$. After substituting $\hat{r}_i$ into (9), the software reliability estimate becomes

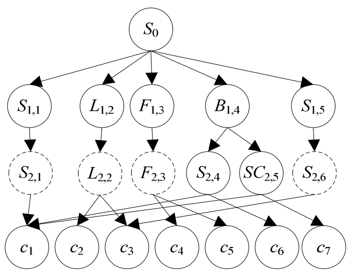
(10)
$$\begin{gathered} \hat{R} = R_{S_0}(R_{S_{1,1}}[R_{S_{2,1}}(\hat{r}_1)], R_{L_{1,2}}[R_{L_{2,2}}(\hat{r}_2, \hat{r}_3)], R_{F_{1,3}}[R_{F_{2,3}}(\hat{r}_4, \hat{r}_5)], \\ R_{B_{1,4}}[R_{S_{2,4}}(\hat{r}_1, \hat{r}_6), R_{SC_{2,5}}(\hat{r}_7, \hat{r}_1)], R_{S_{1,5}}[R_{S_{2,6}}(\hat{r}_3)]). \end{gathered}$$

If the mean of $\hat{r}_i$ (i.e., $\bar{r}_i = E[\hat{r}_i]$) is substituted into (9), the mean of the software reliability estimate can be approximated as

(11)
$$E(\hat{R}) \cong R_{S_0}(R_{S_{1,1}}[R_{S_{2,1}}(\bar{r}_1)],\, R_{L_{1,2}}[R_{L_{2,2}}(\bar{r}_2, \bar{r}_3)],\, R_{F_{1,3}}[R_{F_{2,3}}(\bar{r}_4, \bar{r}_5)],$$
$$R_{B_{1,4}}[R_{S_{2,4}}(\bar{r}_1, \bar{r}_6),\, R_{SC_{2,5}}(\bar{r}_7, \bar{r}_1)],\, R_{S_{1,5}}[R_{S_{2,6}}(\bar{r}_3)]).$$



(a) Connection structure flowchart



(b) The hierarchical model

Fig. 4. An example of component-based software

We assume that the uncertainties of the component reliability estimates are relatively small and the standard deviation, divided by the mean value, is less than 0.3 [8]. We expand (10) at the nominal mean $E(\hat{R})$, using the first-order Taylor series. We obtain

(12)
$$\hat{R} \cong E(\hat{R}) + \sum_{i=1}^{7} b_i(\hat{r}_i - \bar{r}_i),$$

where

$$b_1 = \frac{\partial R_{S_0}}{\partial R_{S_{1,1}}}\frac{\partial R_{S_{1,1}}}{\partial R_{S_{2,1}}}\frac{\partial R_{S_{2,1}}}{\partial r_1} + \frac{\partial R_{S_0}}{\partial R_{B_{1,4}}}\frac{\partial R_{B_{1,4}}}{\partial R_{S_{2,4}}}\frac{\partial R_{S_{2,4}}}{\partial r_1} + \frac{\partial R_{S_0}}{\partial R_{B_{1,4}}}\frac{\partial R_{B_{1,4}}}{\partial R_{SC_{2,5}}}\frac{\partial R_{SC_{2,5}}}{\partial r_1},$$

$$b_2 = \frac{\partial R_{S_0}}{\partial R_{L_{1,2}}}\frac{\partial R_{L_{1,2}}}{\partial R_{L_{2,2}}}\frac{\partial R_{L_{2,2}}}{\partial r_2},\quad b_3 = \frac{\partial R_{S_0}}{\partial R_{L_{1,2}}}\frac{\partial R_{L_{1,2}}}{\partial R_{L_{2,2}}}\frac{\partial R_{L_{2,2}}}{\partial r_3} + \frac{\partial R_{S_0}}{\partial R_{S_{1,5}}}\frac{\partial R_{S_{1,5}}}{\partial R_{S_{2,6}}}\frac{\partial R_{S_{2,6}}}{\partial r_3},$$

$$b_4 = \frac{\partial R_{S_0}}{\partial R_{F_{1,3}}}\frac{\partial R_{F_{1,3}}}{\partial R_{F_{2,3}}}\frac{\partial R_{F_{2,3}}}{\partial r_4},\quad b_5 = \frac{\partial R_{S_0}}{\partial R_{F_{1,3}}}\frac{\partial R_{F_{1,3}}}{\partial R_{F_{2,3}}}\frac{\partial R_{F_{2,3}}}{\partial r_5},\quad b_6 = \frac{\partial R_{S_0}}{\partial R_{B_{1,4}}}\frac{\partial R_{B_{1,4}}}{\partial R_{S_{2,4}}}\frac{\partial R_{S_{2,4}}}{\partial r_6},$$

$$b_7 = \frac{\partial R_{S_0}}{\partial R_{B_{1,4}}}\frac{\partial R_{B_{1,4}}}{\partial R_{SC_{2,5}}}\frac{\partial R_{SC_{2,5}}}{\partial r_7}.$$

All derivatives are evaluated at the nominal value of the component reliability estimates. For example, $(\partial R_{S_0} / \partial R_{S_{1,1}}) = (\partial R_{S_0} / \partial R_{S_{1,1}})|_{(\bar{r}_1,\bar{r}_2,\bar{r}_3,\bar{r}_4,\bar{r}_5,\bar{r}_6,\bar{r}_7)}$. Because

$$R_{S_0} = p_{1,2} p_{3,4} R_{S_{1,1}} R_{L_{1,2}} R_{F_{1,3}} R_{B_{1,4}} R_{S_{1,5}} ,\ R_{S_{1,1}} = R_{S_{2,1}} ,\ R_{L_{1,2}} = R_{L_{2,2}} ,\ R_{F_{1,3}} = R_{F_{2,3}} ,$$

$$R_{B_{1,4}} = p_{4,6} p_{7,10} R_{S_{2,4}} + p_{4,8} p_{8,10} R_{SC_{2,5}} ,\ R_{S_{1,5}} = R_{S_{2,6}} , R_{S_{2,1}} = r_1 ,$$

$$R_{L_{2,2}} = p_{2,3} r_2 r_3 \big/ \left(1 - p_{2,3} p_{3,2} r_2 r_3\right),\ R_{F_{2,3}} = r_4 + r_5 - r_4 r_5 ,\ R_{S_{2,4}} = p_{6,7} r_1 r_6 ,\ R_{SC_{2,5}} = p_{8,9} r_1 r_7 ,$$

$$R_{S_{2,6}} = p_{10,11} r_1^{\ 2} ,$$

the coefficient b_i can be obtained as

$$b_1 = p_{1,2} p_{3,4} \bar{R}_{L_{1,2}} \bar{R}_{F_{1,3}} \bar{R}_{B_{1,4}} \bar{R}_{S_{1,5}} + p_{1,2} p_{3,4} p_{4,6} p_{6,7} p_{8,9} p_{7,10} \bar{R}_{S_{1,1}} \bar{R}_{L_{1,2}} \bar{R}_{F_{1,3}} \bar{R}_{S_{1,5}} \bar{r}_6 +$$

$$p_{1,2} p_{3,4} p_{4,8} p_{8,10} \bar{R}_{S_{1,1}} \bar{R}_{L_{1,2}} \bar{R}_{F_{1,3}} \bar{R}_{S_{1,5}} \bar{r}_7 , b_2 = p_{1,2} p_{2,3} p_{3,4} \bar{R}_{S_{1,1}} \bar{R}_{F_{1,3}} \bar{R}_{B_{1,4}} \bar{R}_{S_{1,5}} \bar{r}_3 \big/ (1 - p_{2,3} p_{3,2} \bar{r}_2 \bar{r}_3)^2 ,$$

$$b_3 = p_{1,2} p_{2,3} p_{3,4} \bar{R}_{S_{1,1}} \bar{R}_{F_{1,3}} \bar{R}_{B_{1,4}} \bar{R}_{S_{1,5}} \bar{r}_2 \big/ (1 - p_{2,3} p_{3,2} \bar{r}_2 \bar{r}_3)^2 + 2 p_{1,2} p_{3,4} p_{9,10} \bar{R}_{S_{1,1}} \bar{R}_{L_{1,2}} \bar{R}_{F_{1,3}} \bar{R}_{B_{1,4}} \bar{r}_1 ,$$

$$b_4 = p_{1,2} p_{3,4} \bar{R}_{S_{1,1}} \bar{R}_{L_{1,2}} \bar{R}_{B_{1,4}} \bar{R}_{S_{1,5}} (1 - \bar{r}_5) ,\ b_5 = p_{1,2} p_{3,4} \bar{R}_{S_{1,1}} \bar{R}_{L_{1,2}} \bar{R}_{B_{1,4}} \bar{R}_{S_{1,5}} (1 - \bar{r}_4) ,$$

$$b_6 = p_{1,2} p_{3,4} p_{4,6} p_{6,7} p_{7,9} \bar{R}_{S_{1,1}} \bar{R}_{L_{1,2}} \bar{R}_{F_{1,3}} \bar{R}_{S_{1,5}} \bar{r}_1 ,\ b_7 = p_{1,2} p_{3,4} p_{4,8} p_{8,9} p_{8,10} \bar{R}_{S_{1,1}} \bar{R}_{L_{1,2}} \bar{R}_{F_{1,3}} \bar{R}_{S_{1,5}} \bar{r}_1 .$$

We rearrange (12) and obtain

$$\left(\hat{R} - E(\hat{R})\right)^2 \cong \sum_{i=1}^{7} b_i^2 (\hat{r}_i - \bar{r}_i)^2 + \sum_{i \neq j}^{7} b_i b_j (\hat{r}_i - \bar{r}_i)(\hat{r}_j - \bar{r}_j) . \tag{13}$$

Taking the expectation of both sides of (13), the variance of the software reliability estimate is

$$\hat{\mathrm{var}}(\hat{R}) \cong \sum_{i=1}^{7} b_i^2 E[(\hat{r}_i - \bar{r}_i)^2] = \sum_{i=1}^{7} b_i^2 \ \hat{\mathrm{var}}(\hat{r}_i) . \tag{14}$$

We notice that only $\bar{r}_i$ and $\hat{\mathrm{var}}(\hat{r}_i)$ are needed for estimating b_i and $\hat{\mathrm{var}}(\hat{R})$. Therefore, our model does not require computation of the higher order moments of the component reliability estimates. Hence, it significantly simplifies the computation steps of $\hat{\mathrm{var}}(\hat{R})$.

4. Experiments and analyses

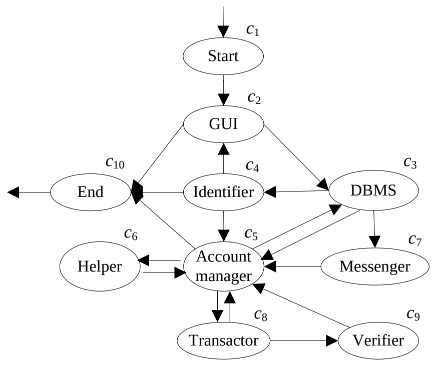
Our experiment has been carried out by using an ATM bank system [9]. The software system structure is shown in Fig. 5a. It consists of ten components and components c_2, c_3, c_5, c_6 and c_9 contains a natural fault respectively. Fig. 5b shows the connection structure flowchart of it.

4.1. Software reliability estimate and comparison of the approaches to variance estimation

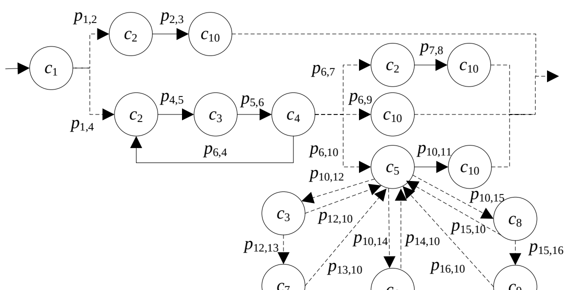
According to these five faults, five versions of the software were constructed and each version contained one fault. We randomly generated inputs to estimate the reliability of each individual faulty component until it was converged. The operational behaviours were collected to calculate the transition probability. c_5 is divided into sub-components $c_{5,1}$ and $c_{5,2}$. The reliability estimates are as follows: $\hat{r}_1$=1.000, $\hat{r}_2$=0.987, $\hat{r}_3$=0.998, $\hat{r}_4$=1.000, $\hat{r}_{5,1}$=1.000, $\hat{r}_{5,2}$=0.996, $\hat{r}_6$=0.994, $\hat{r}_7$=1.000, $\hat{r}_8$=1.000, $\hat{r}_9$=0.976, $\hat{r}_{10}$=1.000. The transition probabilities are shown in Fig. 5b.

For the hierarchical model of ATM bank system, the modules or components at each layer are shown in Table 1. The software reliability estimate is 0.864 according to the approach in Section 3. Due to limitations in space, the detail calculation process is omitted.

The traditional approaches to variance estimation consider the dependence of the components, and need computation of higher order moments. These procedures are often time-consuming. If we assume that the components are *s*-independent, the estimation process will be facilitated, yet the variance of the software reliability estimate will be underestimated [1, 2]. Jin's hierarchical decomposition approach [2] does not need the computation of higher order moments, but it is just for series-parallel systems. Our approach can be suitable for complex structure systems with duplicated components.



(a) System structure



$p_{1,2}$=0.001, $p_{1,4}$=0.999, $p_{2,3}$=1.000, $p_{4,5}$=1.000, $p_{5,6}$=1.000, $p_{6,4}$=0.047,
$p_{6,7}$=0.001, $p_{6,9}$=0.001, $p_{6,10}$=0.951, $p_{7,8}$=1.000, $p_{10,11}$=1.000, $p_{10,12}$=0.436,
$p_{10,14}$=0.100, $p_{10,15}$=0.464,$p_{12,10}$=0.781, $p_{12,13}$=0.219, $p_{13,10}$=1.000,
$p_{14,10}$=1.000, $p_{15,10}$=0.010, $p_{15,16}$=0.990, $p_{16,10}$=1.000.

(b) Connection structure flowchart

Fig. 5. ATM bank system structure and its connection structure flowchart

Table 1. The set of modules or components for each layer

Layer	The set of modules or components
0	$S_0=\{ S_{1,1}, B_{1,2}\}$
1	$S_{1,1}=\{S_{2,1}\}$, $B_{1,2}=\{S_{2,2}, S_{2,3}\}$
2	$S_{2,1}=\{S_{3,1}\}$, $S_{2,2}=\{S_{3,2}\}$, $S_{2,3}=\{L_{3,3}, B_{3,4}\}$
3	$S_{3,1}=\{S_{4,1}\}$, $S_{3,2}=\{S_{4,2}\}$, $L_{3,3}=\{L_{4,3}\}$, $B_{3,4}=\{S_{4,4}, S_{4,5}, S_{4,6}\}$
4	$S_{4,1}=\{S_{5,1}\}$, $S_{4,2}=\{S_{5,2}\}$, $L_{4,3}=\{L_{5,3}\}$, $S_{4,4}=\{S_{5,4}\}$, $S_{4,5}=\{S_{5,5}\}$, $S_{4,6}=\{BC_{5,6}, S_{5,7}\}$
5	$S_{5,1}=\{S_{6,1}\}$, $S_{5,2}=\{S_{6,2}\}$, $L_{5,3}=\{L_{6,3}\}$, $S_{5,4}=\{S_{6,4}\}$, $S_{5,5}=\{S_{6,5}\}$, $BC_{5,6}=\{S_{6,6},B_{6,7}\}$, $S_{5,7}=\{S_{6,8}\}$
6	$S_{6,1}=\{S_{7,1}\}$, $S_{6,2}=\{S_{7,2}\}$, $L_{6,3}=\{L_{7,3}\}$, $S_{6,4}=\{S_{7,4}\}$, $S_{6,5}=\{S_{7,5}\}$, $S_{6,6}=\{S_{7,6}\}$, $B_{6,7}=\{B_{7,7}, S_{7,8}, B_{7,9}\}$, $S_{6,8}=\{S_{7,10}\}$
7	$S_{7,1}=\{S_{8,1}\}$, $S_{7,2}=\{S_{8,2}\}$, $L_{7,3}=\{L_{8,3}\}$, $S_{7,4}=\{S_{8,4}\}$, $S_{7,5}=\{S_{8,5}\}$, $S_{7,6}=\{S_{8,6}\}$, $B_{7,7}=\{S_{8,7}, S_{8,8} \}$, $S_{7,8}=\{S_{8,9}\}$, $B_{7,9}=\{S_{8,10}, S_{8,11}\}$, $S_{7,10}=\{S_{8,12}\}$
8	$S_{8,1}=\{c_1\}$, $S_{8,2}=\{c_2,c_{10}\}$, $L_{8,3}=\{c_2,c_3,c_4\}$, $S_{8,4}=\{c_2,c_{10}\}$, $S_{8,5}=\{c_{10}\}$, $S_{8,6}=\{c_5\}$, $S_{8,7}=\{c_3\}$, $S_{8,8}=\{c_3,c_7\}$, $S_{8,9}=\{c_6\}$, $S_{8,10}=\{c_8\}$, $S_{8,11}=\{c_8,c_9\}$, $S_{8,12}=\{c_{10}\}$
9	$\{c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8, c_9, c_{10}\}$

4.2. Sensitivity analysis

Sensitivity analysis allows insight into the impact of changing the component reliability with respect to the software reliability. It can help to identify the critical components for a resource allocation. The critical point of software reliability R with respect to a component reliability r_i can be defined as $C_{R,i} = \Delta R / \Delta r_i$. The higher valued critical point indicates the critical component.

The critical points of changing the component reliability are shown in Fig. 6. Take the increment of a component reliability from 0.8 up to 0.85 for example, we suppose that the initial reliability of each component is 0.8, and we start to increase each one from 0.8 to 0.85 in turn and observe its impact on the software reliability. We repeatedly change the different increments of each component and present the critical point of it. Fig. 6 shows that as the component reliability r_i increases, the value of $C_{R,i}$ becomes high. The criticality of all components increases sharply, except $C_{R,6}$ and $C_{R,7}$ in a small increment. The critical values of the components c_3, $c_{5,1}$ and $c_{5,2}$ are higher than those of other components and the critical values of the components c_6 and c_7 are lower than those of the other components, which conforms to the actual application of an ATM bank system and is in accordance with the critical points of the software reliability with respect to the component reliabilities in [9]. The software reliability can be improved more efficiently if the critical components are reliable.

5. Conclusions and future work

For the present popular software with duplicated components, an approach to software reliability estimate and its variance estimation for complex structure systems is proposed, which has important applications in software reliability analysis with duplicated components. There is one condition in our approach, that is, the uncertainties of the component reliability estimates must be relatively small. This condition is often satisfied in a risk-averse software design environment or during the manufacturing processes where the parameters often shifted around the nominal ones within a small range. The future work will be focused on the extension of the decomposition method to general software or manufacturing processes, where the condition of small variations is violated.

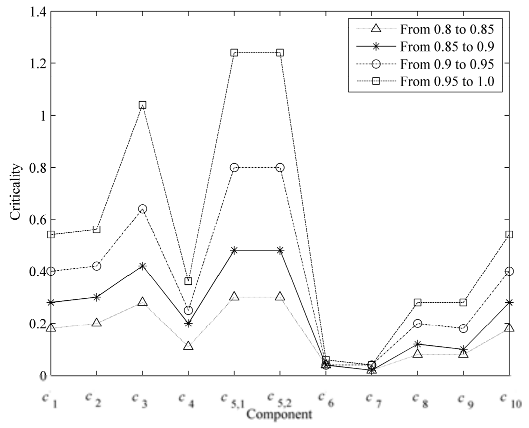


Fig. 6. The criticality of changing the component reliability

Acknowledgement: This paper was supported by the National Natural Science Foundation of China (61170254), Natural Science Foundation of Hebei Province (F2012201145), Science and Technology Research and Development Guidance Plan Project of Baoding City (13ZG012) and the Science Foundation of Hebei University(2013250).

References

1. Jin, T., D. Coit. Variance of System-Reliability Estimates with Arbitrarily Repeated Components. – IEEE Transactions on Reliability, Vol. **50**, 2001, No 4, 409-413.
2. Jin, T. Hierarchical Variance Decomposition of System Reliability Estimates with Duplicated Components. – IEEE Transactions on Reliability, Vol. **57**, 2008, No 4, 564-573.
3. Mohamed, A., M. Zulkernine. A Control Flow Representation for Component-Based Software Reliability Analysis. – In: Proceedings of International Conference on Software Security and Reliability, IEEE, 2008, 1-10.
4. Wang, Q., Y. Lu, Z. Xu, J. Han. Software Reliability Model for Component Interaction Mode. – Journal of Electronics (China), Vol. **28**, 2011, No 4, 632-642.
5. Fiondella, L., S. Rajasekaran, S. S. Gokhale. Efficient Software Reliability Analysis with Correlated Component Failures. – IEEE Transactions on Reliability, Vol. **62**, 2013, No 1, 244-255.
6. Coit, D. System-Reliability Confidence-Intervals for Complex-Systems with Estimated Component-Reliability. – IEEE Transactions on Reliability, Vol. **46**, 1997, No 4, 487-493.
7. Hill, S. D., J. C. Spall, C. J. Maranzano. Inequality-Based Reliability Estimates for Complex Systems. – Naval Research Logistics, Vol. **60**, 2013, No 5, 367-374.
8. Chen, Y., B. Li, C. Yue. Total Sensitivity Index Calculation via Error Propagation Equation. – In: Proceedings of International Conference on Innovative Computing, IEEE, 2007, 619 p.
9. Hsu, C., C. Huang. An Adaptive Reliability Analysis Using Path Testing for Complex Component-Based Software Systems. – IEEE Transactions on Reliability, Vol. **60**, 1997, No 1, 158-170.