高技术通讯 2010 年 第 20 卷 第 4 期:386 ~ 391

doi:10.3772/j.issn.1002-0470.2010.04.010

一个故障检测与故障修正相结合的软件可靠性增长模型①

舒燕君② 刘宏伟 吴智博 杨孝宗

(哈尔滨工业大学计算机科学与技术学院 哈尔滨 150001)

摘 要 针对现有软件可靠性增长模型(SRGM)缺乏故障修正过程的可靠性分析,从故障数量上研究了故障检测过程和故障修正过程的相关性,提出了一个 S 形的相关性函数,并以此建立了一个故障检测与故障修正相结合的非齐次泊松过程类软件可靠性增长模型(RDC-SRGM)。通过实例应用最小二乘法对模型的参数进行了估计。实验结果表明,RDC-SRGM 在故障检测过程和故障修正过程上,均比其它模型的拟合效果和预测能力更好。

关键词 软件可靠性增长模型(SRGM),非齐次泊松过程,故障检测,故障修正

0 引 言

软件可靠性是软件质量的重要指标之一,为了评估和预测软件产品的可靠性,一系列基于非齐次泊松过程(non-homogeneous Poisson process,NHPP)的软件可靠性增长模型(software reliability growth model,SRGM)被相继提出[1],并且已经成为软件可靠性工程实践中非常成功的工具[2-4]。然而,传统的 SRGM 是根据软件测试中的故障检测过程建立的,假设被检测到的故障能够被立即修正,故障的修正过程则被省略。实际上,故障修正是软件测试的主要目的,也是提高软件可靠性的最重要的手段之一。由于缺乏故障修正过程的可靠性分析,传统的 SRGM 不能全面地反映软件测试的情况,这就降低了模型的实际意义和结果的可信度[5]。近期一些学者提出了应当将故障检测与故障修正结合起来建立 SRGM 的观点。通过直接使用传统的 SRGM 作为故障检测过程模型,从时间延迟的角度分析故障检测过程和故障修正过程之间的相关性,并进一步建立故障修正过程的模型。其中,Schneidewind[5, 6]首先提出应当对故障修正过程建立模型,认为故障修正过程可以看作是一个固定时间延迟的故障检测过程,通过使用最早出现的 SRGM——GO(Goel and Okumoto, GO)模型作为故障检测过程的模型,故障修正过程模型则是固定常数型延迟的故障检测过程模型。Xie[7, 8]在 Schneidewind 的基础上将常数型的时间延迟替换为函数型的时间延迟。他们认为随着测试的进行,时间延迟较长的故障占软件中潜伏故障总数的比例会提高,因而时间延迟将不断增长。Shibata[9], Huang[10]等假设故障的检测和修正是一个排队系统,已检测故障等待进一步修正的过程可以看作是进入服务队列等待服务的过程,并以此为基础建立了故障修正过程模型。Kapur[11], Lo[12]等假设故障的检测和修正均服从 NHPP 类过程,根据"先检测,后修正"的特点建立了结合故障检测与故障修正的基本框架。

现有的一些结合故障检测与故障修正的 SRGM 通常是从时间的角度去分析两个过程之间的关系,这些关系通常由研究者根据自身的测试经验假设得出,因此模型缺乏事实依据。实际上,故障检测与故障修正在故障数量上也是相关的。通过分析实际故障数据集可以发现,故障检测个数与故障修正个数的变化趋势几乎一致。因此,从故障数量的角度将故障检测与故障修正相结合是更直接、更有效的方法。本文从软件测试的实际数据出发,分析了故障检测与故障修正在故障数量上的相关性,建立了一种新的故障检测与故障修正相结合(relating the detection with the correction, RDC)的 SRGM——RDCSRGM,该模型能够较好地对软件测试的故障检测过程和故障修正过程进行可靠性分析。

① 863 计划(No.2008AA01A201)和国家自然科学基金(No.60503015)资助项目。
② 舒燕君,女,1981 年生,博士研究生,讲师;研究方向:容错计算,软件测试;联系人,E-mail:yjshu@hit.edu.cn
(收稿日期:2008-12-18)

1 NHPP 类 SRGM

注释:

$N(t)$: $t \geqslant 0$, 一个计数过程,表示到时刻 t 检测到的累积故障数;

a :软件系统中潜伏的故障总数的初始值;

$a(t)$:基于时间的潜伏故障总数的函数,即到时刻 t,已修正的软件故障数和潜伏在软件中尚未被发现的软件故障数的和;

$b(t)$:基于时间的故障检测率函数,即软件中每个故障在时刻 t 被检测到的平均概率;

$m(t)$:到时刻 t 为止能够发现的累积故障数的期望值。

NHPP 类型的 SRGM 用一个计数过程 $\{N(t), t \geqslant 0\}$ 表示到时刻 t 为止检测到的软件故障累计数。如果故障累计数的期望值函数用 $m(t)$ 表示,则一个基于 NHPP 过程的 SRGM 的一般形式为

$$P\{N(t) = n\} = \frac{m(t)^n}{n!} e^{-m(t)} \tag{1}$$

NHPP 类 SRGM 通常假设故障检测率 $b(t)$ 与软件中存在的故障成正比,Pham[13] 等在此假设的基础上提出了建立 NHPP 类 SRGM 的通用框架,该框架的形式为

$$\frac{\mathrm{d}m(t)}{\mathrm{d}t} = b(t) \cdot [a(t) - m(t)] \tag{2}$$

解方程(2)可得到一个 NHPP 类 SRGM 的通用形式

$$m(t) = \exp(-B(t)) \cdot \left[m_0 + \int_0^t a(\tau) b(\tau) \exp(B(\tau)) \mathrm{d}\tau \right] \tag{3}$$

其中, $B(t) = \int_0^t b(\nu)\mathrm{d}\nu$, $m_0 = m(t_0)$, t_0 表示测试开始的时间。

当 $a(t) = a$, $b(t) = b$ 时,方程的解为 $m(t) = a(1 - e^{-bt})$, 即 GO 模型。GO 模型是最基本的 NHPP 类 SRGM,在实际的测试中有着十分广泛的应用,后来的很多 SRGM 都是对 GO 模型的假设条件修改得到的[2, 3]。

传统的 SRGM 中 $m(t)$ 只能对软件测试的故障检测过程进行可靠性分析。在实际的软件工程中,被检测到的故障需要经过故障分析、故障定位以及修改代码之后才能被修正。故障修正过程是软件测试的重要组成部分,它直接影响着软件的可靠性增长速度,决定了软件的发布时间、测试资源的分配。因此,需要在软件可靠性建模中将故障检测过程和故障修正过程结合起来,才能够得到全面反映实际测试情况的 SRGM,为测试人员评测软件可靠性、确定软件发布时间提供更多的依据。

2 故障检测和故障修正在故障数量上的相关性研究

近期,一些研究者提出了故障检测与故障修正相结合的 SRGM。他们直接使用传统的 SRGM 作为描述故障检测过程的模型,即 $m(t) = m_d(t)$,以此为基础分析故障修正和故障检测之间的关系,并进一步建立故障修正过程的模型 $m_c(t)$。

从引言中对现有结合故障检测与故障修正的 SRGM 介绍可以看出,这些模型都是从时间的角度建立。在软件的测试过程中,从故障的检测到修正所需要的时间受到多种因素的影响,包括软件系统的规模、软件结构的复杂度、已检测到的故障个数和故障修正人员对软件测试工具的熟练程度等。因此,从时间的角度并不能准确地描述故障检测过程和故障修正过程之间的关系,正确反映故障修正过程对软件可靠性增长带来的影响,需要有一种更加实际有效的方法将两个过程结合起来。

实际上,在软件测试中,故障在检测到之后才可能被修正,修正故障的数量会随着检测故障的数量变化,所以可以从故障数量上来分析两个过程之间的关系。图 1(a)和图 1(b)中分别给出了 Xie 的中等规模软件项目[13]和 Musa 的 T1 系统[10]在软件测试中的故障检测过程与故障修正过程的故障数变化。从图 1 的两个实例可以看出,随着测试的进行,检测故障数与修正故障数的变化趋势非常相似,因此可以直接从故障数量上来分析这两个过程之间的相关性。

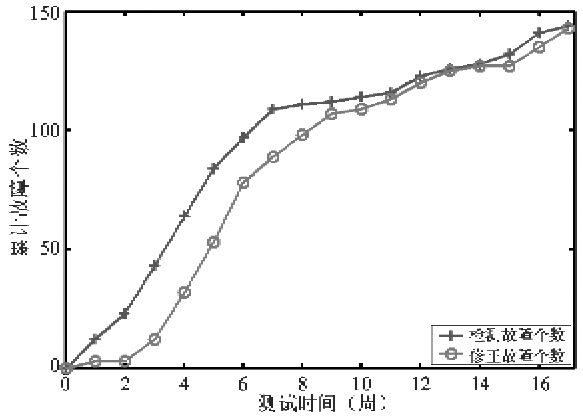
在本文中,我们通过使用修正故障个数和检测故障个数的比值来分析这两个过程之间的相关性。假设相关性函数 $r(t)$ 代表了修正故障个数和检测故障个数的比值,即: $r(t) = m_c(t)/m_d(t)$, $0 < r(t) \leqslant 1$。图 2(a)和图 2(b)中的点分别表示 Xie 的软件项目和 Musa 的 T1 系统在实际测试中的 $r(t)$ 值。从图 2 中可以看出,故障检测与故障修正在故障数量的比值上呈非递减的 S 形曲线。下面,我们进一步对 $r(t)$ 的这种变化趋势进行分析。

在软件测试的初期,软件中存在的故障较多,故障被检测出的速度较快。然而,排错人员对于软件系统的结构不熟悉,对测试工具的使用不熟练,故障修正过程滞后于故障检测过程, $r(t)$ 远小于 1。随

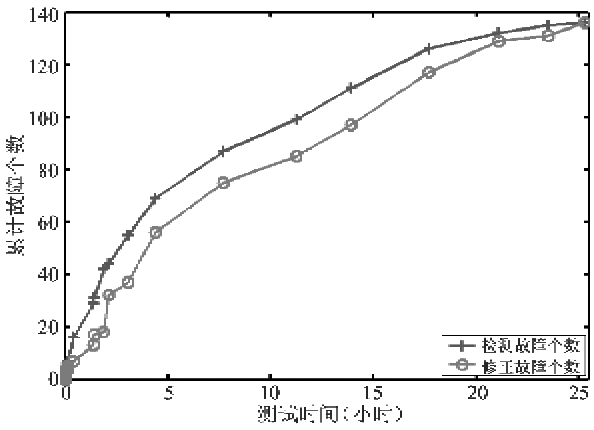
着软件测试过程的进行,检测故障个数的增长速度减慢。另一方面,排错人员对软件结构和测试工具逐渐掌握,能够较快地对已检测到故障进行分析和定位,并在较短时间内将故障修正,因此故障修正过程将快速接近于故障检测的过程,此时 $r(t)$快速增长。在测试的末期,由于软件中大部分的故障都已被检测和修正,故障修正过程逐步接近于故障检测过程,$r(t)$也几乎接近于 1。相关性函数 $r(t)$的这种 S 形的非递减趋势可以用罗吉曲线(logistic curve)表示[14],罗吉曲线在实际应用中有很多优秀表现,如式

$$r(t) = \frac{1}{1 + \eta e^{-b \cdot t}} \tag{4}$$

所示。图 2(a)和图 2(b)中的虚线是使用 $r(t)$根据实际的比值进行估计得出的。从图 2 可以看出,相关性函数 $r(t)$所估计得出的曲线十分吻合修正故障个数和检测故障个数的实际比值。

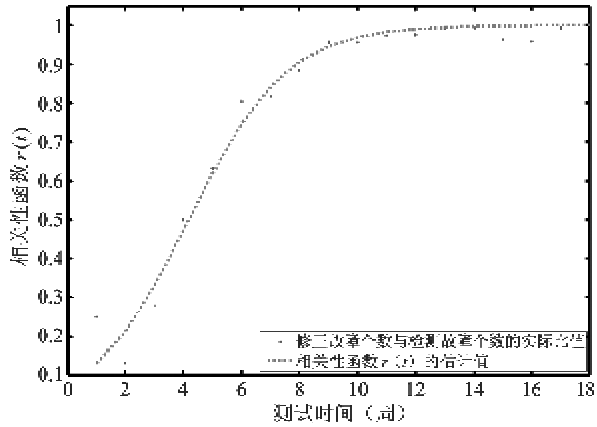


(a)一个中等规模软件

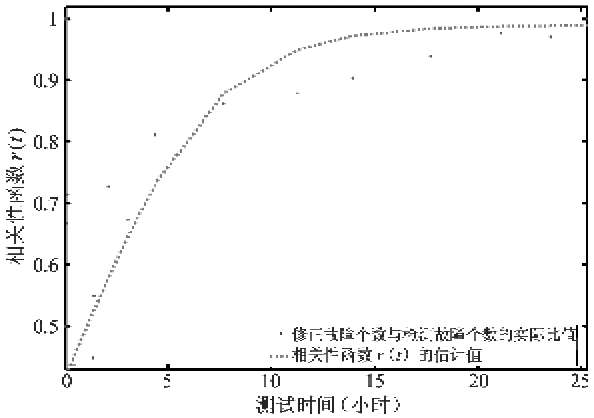


(b)T1 系统

图 1 实际检测故障数与修正故障数的比较



(a)一个中等规模软件



(b)T1 系统

图 2 故障数实际比值与 $r(t)$的比较

3 一个故障检测与故障修正相结合的 SRGM

本节建立一个基于 NHPP 过程的故障检测与故障修正相结合的 SRGM,该模型能够对软件测试中的故障检测过程和故障修正过程进行可靠性分析和预测。NHPP 类 SRGM 普遍具有如下假设[2, 3]:

(1)软件的失效遵循 NHPP 过程;

(2)在任何时刻软件的失效都是由软件中存在的故障引起的;

(3)在任何时间序列 $t_0 < t_1 < \ldots < t_n$ 构成的时间区间 $[t_0, t_1), [t_1, t_2), \cdots, [t_{n-1}, t_n)$ 中检测到的故障数是相互独立的;

(4)在任何时候软件的失效强度与软件中的残留故障成正比。

根据假设,可以得到

$$\frac{\mathrm{d}m_d(t)}{\mathrm{d}t} = b(t)[a(t) - m_c(t)] \tag{5}$$

通过上一小节在两组实际数据上的实验分析可

知,故障检测过程与故障修正过程之间的相关性函数可以表示为

$$r(t) = \frac{m_c(t)}{m_d(t)} = \frac{1}{1 + \eta e^{-\partial \cdot t}} \tag{6}$$

式(6)可以转化为

$$m_c(t) = r(t) \cdot m_d(t) = \frac{1}{1 + \eta e^{-\partial \cdot t}} \cdot m_d(t) \tag{7}$$

将式(7)代入式(5)中,可以得到一个关于$m_d(t)$的导数方程:

$$\frac{\mathrm{d}m_d(t)}{dt} = b(t)\left[a(t) - \frac{1}{1 + \eta \cdot e^{-\partial \cdot t}} \cdot m_d(t)\right] \tag{8}$$

为了便于模型的求解,我们可以进一步假设$a(t)$,$b(t)$为常数,即$a(t) = a$, $b(t) = b$,那么式(8)可以简化为

$$\frac{\mathrm{d}m_d(t)}{\mathrm{d}t} = b\left[a - \frac{1}{1 + \eta \cdot e^{-\partial \cdot t}} \cdot m_d(t)\right] \tag{9}$$

在边界条件$m_d(0) = 0$下,解方程(9)可以得出故障检测过程的均值函数为

$$\begin{aligned} m_d(t) = {} & a \cdot \left[1 - \left(\frac{1 + \eta}{1 + \eta e^{-\partial t}}\right)^{b/\partial} e^{-bt}\right] \\ & - \frac{ab}{\partial}\left[\ln(1 + \eta e^{-\partial t})\right. \\ & \left. - \ln(1 + \eta)\left(\frac{1 + \eta}{1 + \eta e^{-\partial t}}\right)^{b/\partial} e^{-bt}\right] \end{aligned} \tag{10}$$

由式(7)和式(10)可以得到故障修正过程的均值函数为

$$\begin{aligned} m_c(t) = {} & \frac{a}{1 + \eta e^{-\partial t}}\left[1 - \left(\frac{1 + \eta}{1 + \eta e^{-\partial t}}\right)^{b/\partial} e^{-bt}\right] \\ & - \frac{ab}{\partial(1 + \eta e^{-\partial \cdot t})}\left[\ln(1 + \eta e^{-\partial \cdot t})\right. \\ & \left. - \ln(1 + \eta)\left(\frac{1 + \eta}{1 + \eta e^{-\partial t}}\right)^{b/\partial} e^{-bt}\right] \end{aligned} \tag{11}$$

由此,得出了可以故障检测与故障修正相结合的SRGM,我们称之为RDC-SRGM。当$\eta = 0$时,故障的修正过程被忽略,即$m_d(t) = m_c(t) = a(1 - e^{-bt})$,均为GO模型形式。

4 实验分析

本小节使用一个中型软件系统[7]的测试数据集对RDC-SRGM模型的性能进行分析比较,该数据集记录了软件测试中的故障检测过程和故障修正过程。在测试结束时,检测出的故障数为144,修正的故障数为143。此外,用于比较的SRGM包括Schneidewind模型[5],Xie模型[7],Lo模型[12],Huang模型[10]。

4.1 比较标准

用于模型拟合能力的标准有误差平方和(sum of square error, SSE)[8];用于模型适合描述数据程度的标准有K-S距离(Kolmogorov-Smirov distance)[11];用于模型预测能力的标准有相对误差(relative error, RE)[12]和相对误差均值(mean of relative error, MRE)[9]。

假设$t_0 < t_1 < t_2 < \ldots < t_n$表示失效时间,$\{t_i, y_i\}$表示失效间隔数据,$y_i$表示到时间$t_i$为止的实际累计故障数,对上述标准作如下定义:

(1)误差平方和为

$$\mathrm{SSE} = \sum_{k=1}^{n}(y_k - \hat{m}(t_k))^2 \tag{12}$$

*SSE*反映了估计值与实际值之间的偏差。其中,n表示样本数量,$m(t_i)$是到t_i时刻为止故障累积数的估计值。显然SSE越小,表明拟合的误差越少。

(2)K-S距离为

$$D_k = Sup_k |F^*(x) - F(x)| \tag{13}$$

Kolmogorov-Smirov test(柯尔莫诺夫-斯米尔诺夫检验)可以分析两组分布之间有无显著性差异,因此K-S距离可以反映模型适合描述数据程度。其中,k是样本数量,$F^*(x)$,$F(x)$分别代表在x-点标准化之后的实际数据的分布和基于模型期望值的分布,D_k是两个分布之间的最大距离。两组数据的分布越接近,K-S距离越小,表明模型越适用。

(3)相对误差为

$$RE = \frac{\hat{m}(t_Z) - Z}{Z} \tag{14}$$

*RE*反映了模型的预测能力[15],假设在软件测试结束时t_q观测到y_q个软件故障,使用失效数据t_e($t_e \leqslant t_q$)估测SRGM的均值函数$m(t)$的参数,然后将这些参数值代入到$m(t)$中,可以得到软件测试结束时间t_q的累积失效数的估计值。得到的估计值与实际值进行比较,重复不同的失效时间间隔t_e。*RE*越趋近于0,表明模型的预测效果越好。

此外,相对误差均值为*RE*绝对值在整个测试过程的平均值。

4.2 模型评估比较

应用最小二乘法分别对Schneidewind模型,Xie模型,Lo模型,Huang模型和RDC-SRGM模型的参数进行估计,其结果如表1所示。然后根据所估算出的参数值,可以计算出各模型在故障检测过程和故

障修正过程上的比较标准值，如表 2 所示。从表 2 可以看出，RDC-SRGM 模型除了在故障修正过程上 *MRE* 值略大于 Schneidewind 模型之外，其它的比较标准值都最小，说明该模型的总体表现最好。下面对各个模型的比较结果作进一步的分析。

表 1 模型参数估计结果

模型	参数估计值
Schneidewind 模型	$a=153.01$，$b=0.1487$，$c=1.9390$
Xie 模型	$a=168.36$，$b=0.1193$，$c=0.0279$
Lo 模型	$a=156.34$，$b=0.1404$，$\mu=0.5811$
Huang 模型	$a=152.45$，$b=0.2286$，$c=0.5517$，$\mu=0.2519$
RDC-SRGM 模型	$a=160.01$，$b=0.1078$，$\eta=2035$，$\partial=1.7724$

由表 2 可以得出 Schneidewind 模型、Lo 模型、RDC-SRGM 模型在故障检测过程和故障修正过程上的总体拟合结果要好于其他模型，图 3 和图 4 分别是这 3 个模型在故障检测过程和故障修正过程上的估计值和实际故障数据的比较。从图 3 可以看出，Schneidewind 模型、Lo 模型、RDC-SRGM 模型都能比较准确地估计故障检测过程的变化。而从图 4 中可以看出，RDC-SRGM 模型能够比其他两个模型更准确地估计故障检测过程的变化。

由表 2 还可以得出在故障检测过程上 Xie 模型、Lo 模型、RDC-SRGM 模型的 MRE 值比其他模型更小，而在故障修正过程上 Schneidewind 模型、Huang 模型、RDC-SRGM 模型的 MRE 值比其他模型要小。图 5 是在故障检测过程上 MRE 值最小的 3 个模型的 *RE* 值变化趋势，可以看出，RDC-SRGM 模型的 *RE* 值比其它模型要更稳定地接近于 0，表明该模型在故障检测过程上的预测能力比其它两个模型要好。图 6 是 Schneidewind 模型、Huang 模型、RDC-SRGM 模型在故障修正过程上的 *RE* 变化。从图 6 可以看出，在测试的中后期（第 10 周之第 17 周），RDC-SRGM 模型和 Schneidewind 模型的 *RE* 值都十分接近于 0，表明这两个模型在故障修正过程中后期的预测能力十分接近，都要好于 Huang 模型。然而 RDC-SRGM 模型在测试初期 *RE* 值比 Schneidewind 模型更迅速的趋近于 0，说明 RDC-SRGM 模型在故障修正过程初期的预测能力比 Schneidewind 模型要好。

从以上表格和图形可以得到下述结论：在该软件测试数据集上，RDC-SRGM 模型比其他几个模型在故障检测过程和故障修正过程上的拟合效果和预测能力要好。

表 2 模型比较结果

模型	SSE		K-S 距离		MRE	
	检测过程	修正过程	检测过程	修正过程	检测过程	修正过程
Schneidewind 模型	884.17	510.00	0.1064	0.1283	0.1396	0.1879
Xie 模型	987.02	2578.90	0.1568	0.1303	0.1228	0.7358
Lo 模型	861.22	1015.24	0.1203	0.1225	0.1324	0.4435
Huang 模型	1682.60	2189.20	0.1225	0.1662	0.1938	0.2094
RDC-SRGM 模型	667.90	411.70	0.0895	0.0795	0.0989	0.1894

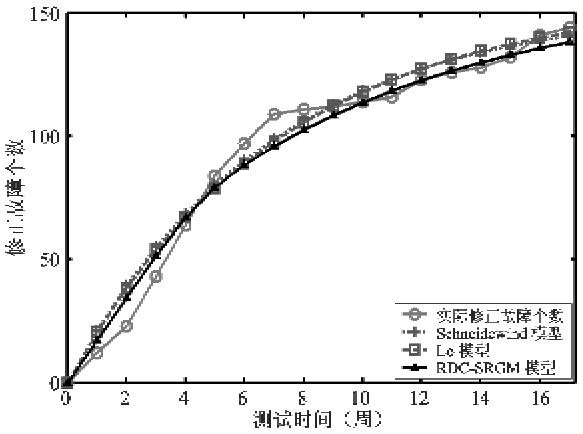


图 3 故障检测的实际个数与模型估计值的比较

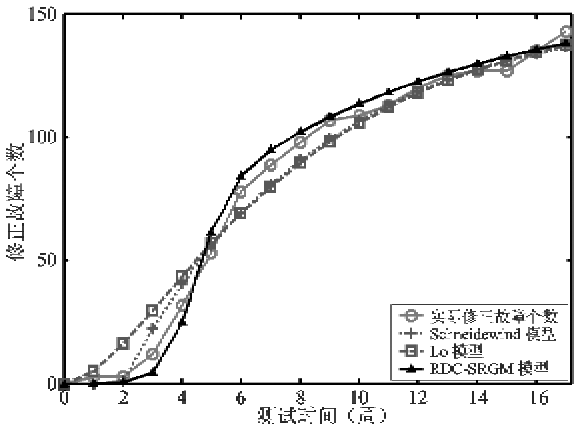


图 4 故障修正的实际个数与模型估计值的比较

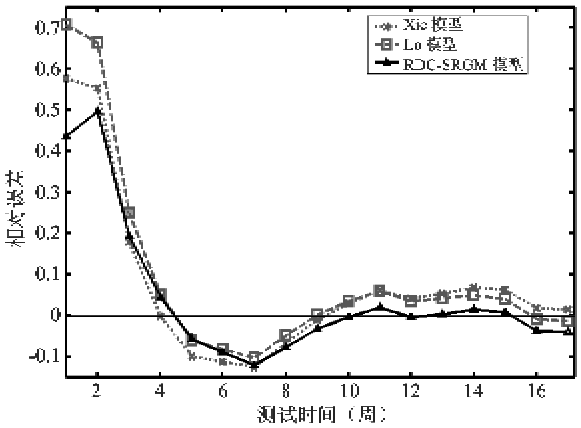


图 5　故障检测过程的 *RE* 值比较

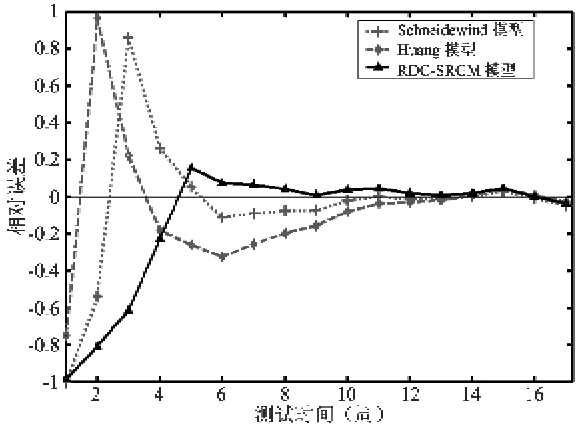


图 6　故障修正过程的 *RE* 值比较

5　结　论

故障修正是提高软件可靠性的重要手段,决定着软件的发布时间和测试资源的分配,因此需要建立能够对故障修正过程进行可靠性分析的软件可靠性增长模型(SRGM)。本文通过分析故障检测过程与故障修正过程在故障数量上的相互关系,得出一个具有 S 形变化趋势的相关性函数,并以此为基础建立了故障检测与故障修正相结合的 SRGMRDC-SRGM。通过公开发表的数据集对模型的拟合效果和预测能力进行了评估,实验结果表明,RDC-SRGM 模型在故障检测过程和故障修正的过程上的拟合效果和预测能力较好,能够比较准确地评估软件可靠性。

参考文献

[1] Lyu M, editor. Handbook of Software Reliability Engineering. New York: McGraw-Hill and IEEE Computer Society, 1996. 25-38

[2] Musa J, Iannino A, Okumoto K. Software Reliability Measurement, Prediction and Application. New York: McGraw-Hill, 1987. 62-78

[3] Ohba M. Software reliability analysis models. *IBM J Research & Development*, 1984, (28): 328-443

[4] 赵靖,刘宏伟,崔刚等. 考虑测试环境和实际运行环境的软件可靠性增长模型. 计算机研究与发展, 2006, 43(5): 881-887

[5] Schneidewind N. Analysis of error processes in computer software. *Signlan Notice*, 1975, (10): 337-346

[6] Schneidewind N. Fault correction profiles. In: Proceedings of the 14th International Conference on Software Reliability Engineering, Denver, USA, 2003, 257-267

[7] Xie M, Zhao M. The Schneidewind software reliability model revisited. In: Proceedings of the 3rd International Symposium on Software Reliability Engineering, Los Alamitos, CA, 1992, 184-192

[8] Xie M, Hu Q, Wu Y, et al. A study of the modeling and analysis of software fault-detection and fault-correction processes. *Journal of Quality and Software Reliability Engineering International*, 2007, (23): 459-470

[9] Shibata K, Rinsaka K, Dohi T, et al. Quantifying Software Maintainability Based on a Fault-Detection Correction Model. In: Proceedings of 13th *IEEE* International Symposium on Pacific Rim Dependable Computing, Melbourne, Australia, 2007, 35-41

[10] Huang C, Huang W. Software reliability analysis and measurement using finit and infinit queueing model, *IEEE Transaction on Reliability*, 2008, 57(1): 192-203

[11] Kupar P, Goswami D, Badham A et al. Flexible software reliability growth model with testing effort dependent learning process. *Applied Mathmatical Modeling*, 2008, (32): 1298-1307

[12] Lo J, Huang C. An integration of fault detection and correction processes in software reliability analysis. *Journal of System and Software*, 2006, (29): 1312-1323

[13] Pham H, Nordman L, Zhang X. A general imperfect software debugging model with S-shape fault detection rate. *IEEE Transaction on Reliability*, 1999, 48(2): 169-175

[14] Huang C, Kuo S. Analysis of incorporating logistic testing-effort function into software reliability modeling. *IEEE Transactions on Reliability*, 2002,51(3): 261-270

A software reliability growth model integrating fault detection and fault correction processes

Shu Yanjun, Liu Hongwei, Wu Zhibo, Yang Xiaozong

(School of Computer Science & Technology, Harbin Institute of Technology, Harbin 150001)

Abstract

Aiming at the problem that traditional software reliability growth models (SRGMs) only deal with the fault detection process and ignore the fault correction process, this paper studies the relationship between these two processes from the viewpoint of fault amount and presents an S-shaped function to characterize it. Based on this function, the RDC-SRGM, a SRGM of the kind of the non-homogeneous Poisson process (NHPP), which integrates fault detection and fault correction, is proposed. Finally, the unknown parameters are estimated by using the least square estimation method based on a software testing data set. The experimental results show that the RDC-SRGM has the better goodness-of-fit ability and predictive power than other models on both fault detection and correction processes.

Key words: software reliability growth model(SRGM), non-homogeneous Poisson process, fault detection, fault correction