

考虑测试工作量和故障检测率的软件可靠性模型

周波, 雷航, 郭文生

(电子科技大学 信息与软件工程学院, 成都 610054)

摘要: 针对软件可靠性模型中对故障检测率行为描述的不足, 考虑软件固有的故障检测率是一个随时间增加的减函数, 以及测试人员的学习能力是一个随时间增加呈 S 型的增函数, 两者共同决定了故障检测率是一个随时间增加的先增后减的函数, 于是提出改进的软件可靠性模型。在改进模型的基础上, 考虑测试工作量对可靠性的影响, 软件可靠性模型得到了进一步的改善。利用公开发表的失效数据集对改进的模型进行比较和验证, 实验数据证明改进的模型具有很好的效果。

关键词: 软件可靠性模型; 故障检测率; 测试工作量; 学习能力; 模型比较

中图分类号: TP393.04

文献标志码: A

文章编号: 1001-3695(2016)07-2071-04

doi:10.3969/j.issn.1001-3695.2016.07.033

Software reliability model considering testing effort and fault detection rate

Zhou Bo, Lei Hang, Guo Wengsheng

(School of Information & Software Engineering, University of Electronic Science & Technology of China, Chengdu 610054, China)

Abstract: The behavior description of the fault detection rate was inadequate. This paper considered the inherent fault detection rate and testers' learning ability. The inherent software fault detection rate changing with time was a decreasing function. Learning ability of the testers increased as S-shaped type with time changing. The fault detection rate increased first and then decreases with time. This paper proposed an improved model. On the basis of the improved model, this paper considered the impact of the testing effort and then proposed a new testing effort function and a software reliability model. The reported data set used to test the improved model and compared with the other existing models. The experimental results show that the improved models have a good effect.

Key words: software reliability model; fault detection rate; testing effort; learning ability; model comparison



软件可靠性 (software reliability) 是系统可依赖性的关键因素, 是最重要的软件质量特性之一。IEEE 计算机学会把软件可靠性定义为在规定的条件下, 在规定时间内, 软件不发生失效的概率^[1]。经过近四十年的研究大量的软件可靠性模型被提出, 非齐次泊松过程 (non-homogeneous poisson process, NHPP) 类软件可靠性模型是其中很重要的一类模型, 目前对于这类模型的研究非常活跃并且已经有很多运用到实际的成功案例。Goel-Okumoto 软件可靠性模型^[2] (G-O 模型) 是在 1973 年由 Goel 和 Okumoto 提出。它是 NHPP 类软件可靠性模型中的经典模型之一。在 G-O 模型的基础上提出了很多新的模型, 例如 Yamada Delayed S-Shaped 模型和 Inflected S-Shaped 模型^[3,4] 等等。

在整个软件的测试过程中软件测试工作量被认为是一种不可忽视的因素^[5], 测试工作量是指在测试过程中所消耗的各种测试资源, 它可以利用测试过程中所需的人力、测试用例数的多少、CPU 时间等信息来进行度量^[6]。最终的软件可靠性水平和测试工作量之间联系非常紧密^[7], 测试工作量对软件可靠性有明显的积极作用^[6,7]。因此将测试工作量函数 (testing effort function, TEF) 引入软件可靠性增长模型中能够提高软件可靠性的评估能力。

文献[8]中提出不完美排错和修正故障中新故障的引入

无须重点考虑, 因为两者在模型的精确性上影响不是很大。因此本文在不考虑不完美排错和新故障的引入的情况下对 NHPP 类软件可靠性模型进行改进。在改进的模型的基础上考虑测试工作量的因素进一步改进模型, 并利用公开的数据集来对本文中的模型进行测试, 并且和已有的多种模型进行比较, 以证明本文中的模型的有效性。

1 传统的 NHPP 模型

NHPP 模型是以单位时间内的失效次数为独立泊松随机变量的泊松类模型。软件故障的发现被认为是一个计数过程, 它可以用 $\{N(t), t \geq 0\}$ 来表示并服从非齐次泊松分布, 密度函数为 $\lambda(t)$, 均值函数为 $m(t)$ 。

$$m(t) = \int_0^t \lambda(s) ds \quad (1)$$

由非齐次泊松分布过程可以得到

$$P\{N(t) = n\} = \frac{[m(t)]^n}{n!} e^{-m(t)}, n = 0, 1, 2, \dots \quad (2)$$

Hoang Pham 等人提出了一个 NHPP 类模型的框架, 其框架形式为

$$\frac{dm(t)}{dt} = b(t) \times (a(t) - m(t)) \quad (3)$$

其中: $b(t)$ 为故障发现率函数, $a(t)$ 为软件的总故障函数。以

收稿日期: 2015-06-16; 修回日期: 2015-08-06

作者简介: 周波 (1990-), 男, 硕士研究生, 主要研究方向为软件可靠性测试与评估 (zhoubo_760@163.com); 雷航 (1960-), 男, 教授, 博士, 主要研究方向为软件可靠性工程; 郭文生 (1976-), 男, 副教授, 博士, 主要研究方向为软件测试技术、计算机网络及应用。

GO 模型为例 $b(t)$ 和 $a(t)$ 都为常数, 分别为 b 和 a 。再利用式 (3) 可解得

$$m(t) = a \times (1 - \exp(-b \times t)) \quad (4)$$

软件失效强度 $\lambda(t)$ 是指单位时间内软件失效的机会或可能。

$$\lambda(t) = \frac{dm(t)}{dt} = a \times b \times \exp(-b \times t) \quad (5)$$

$R(x|t)$ 表示在时间间隔 $(t, t+x)$ 内软件不发生故障的概率, t 为软件最后一次发生故障的时刻。在 G-O 模型中软件可靠度为

$$R(x|t) = P\{N(t+x) - N(t) = 0\} = \exp(m(t) - m(t+x)) \quad (6)$$

2 考虑测试工作量和故障检测率的 NHPP 建模方法

根据文献[6]给出以下假设:

- a) 在 0 时刻没有错误发生;
- b) 软件发生失效的过程是一个 NHPP 过程;
- c) 软件的错误出现后, 错误会被立即改正, 不会引入新的错误, 每次只修正一个错误;
- d) 每一个时间间隔内发现的错误数是相互独立的;
- e) 在测试时间间隔 $(t, t+h)$ 内检测到错误的期望值和当前测试工作量消耗率与此时的残留错误数成比例。

根据假设条件可以得到如式 (7) 所示的考虑测试工作量和故障检测率的 NHPP 软件可靠性建模框架。

$$\frac{dm(t)}{dt} = w(t) \times b(t) \times (a - m(t)) \quad (7)$$

$$w(t) = \frac{dW(t)}{dt} \quad (8)$$

其中: $m(t)$ 是在时间间隔 $[0, t]$ 内的检测到的错误数的期望均值函数; $w(t)$ 表示测试工作量消耗率函数; $W(t)$ 表示测试工作量函数。

假设 $B(t) = \int_0^t b(W(\tau)) d\tau$, 在边界条件为 $m(0) = 0$, $W(0) = 0$ 时, 由式 (7) 可获得考虑测试工作量和故障检测率的形式化 NHPP 模型为

$$m(t) = ae^{-B(W(t))} \int_0^t e^{B(W(\tau))} b(W(\tau)) d\tau \quad (9)$$

3 故障检测率函数

本文中假设故障检测率是一个随时间变化的函数。在实际过程中故障检测率由下面两个因素共同决定:

- a) 软件中的故障随时间增加不断被排出, 剩余故障数不断减少, 故障越难发现, 因此故障检测率不断减小。
- b) 测试人员的学习能力^[9,10]对故障检测率会产生积极的影响, 在软件测试过程中测试人员的检测故障的能力会随时间的增加而提高, 它是一个与时间相关的递增的函数。

测试阶段软件固有的故障检测率 $b_1(t)$ 的变化趋势为

$$b_1(t) = \exp(-c \times t) \quad (10)$$

测试人员在测试过程中的学习能力是呈增长趋势, 它是一个非递减函数, 函数表示为

$$b_2(t) = \frac{1 - \exp(-c \times t)}{1 + d \times \exp(-c \times t)} \quad (11)$$

d 和 c 都是相关的参数。

因此故障检测率 $b(t)$ 由式 (10) (11) 共同决定表达如下:

$$b(t) = b_1(t) \times b_2(t) = \frac{\exp(-c \times t) \times (1 - \exp(-c \times t))}{1 + d \times \exp(-c \times t)} \quad (12)$$

由 $m(0) = 0, a(t) = a$, 根据式 (3) (12) 解得

$$m(t) = a \times (1 - \exp(\frac{1}{c \times d} - \frac{\ln(d+1) \times (d+1)}{c \times d^2})) \times \exp(\frac{\ln(d + \exp(c \times t)) \times (d+1)}{c \times d^2} - \exp(-c \times t) \times (\frac{1}{c \times d} + \frac{t \times \exp(c \times t) \times (d+1)}{d^2})) \quad (13)$$

4 测试工作量函数

1) 变形 S 型测试工作量函数

文献[11]利用变形 S 型 (inflected S-shaped) 函数来描述测试工作量的 S 型增长趋势, 它有灵活性好、结构简单等优点。在 $[0, t]$ 内消耗的累积测试工作量函数如式 (14) 所示。

$$W(t) = W_{\max} \frac{1 - e^{-rt}}{1 + \psi e^{-rt}} \quad (14)$$

其中: r 表示测试工作量的消耗率; ψ 是一个常量; $W_{\max}$ 表示可消耗的测试工作量的最大值。由式 (14) 可知: a) $W(0) = 0$, 即初始工作量为 0; b) $W(t)$ 为正且随时间 t 单调递增, 这与测试过程中测试工作量不断增加的过程相同; c) 对 $W(t)$ 进行求导, 则可以得到测试工作量增长速率函数 $w(t)$ 为

$$w(t) = \frac{dW(t)}{dt} = \frac{W_{\max} r \exp(-rt) (1 + \psi)}{(1 + \psi \exp(-rt))^2} \quad (15)$$

2) LTCS 测试工作量函数

假设利用文献[12]中的 LTCS 软件可靠性模型来描述测试工作量随时间的变化情况, 则 $[0, t]$ 内的累积测试工作量函数如式 (16) 所示。

$$W(t) = W_{\max} (1 - \exp(-\frac{\alpha t}{1 + \psi \exp(-\alpha t)})) \quad (16)$$

其中: α 表示测试工作量的消耗率; ψ 是一个常量; $W_{\max}$ 表示可消耗的测试工作量的最大值。由式 (16) 可知: a) $W(0) = 0$, 即初始工作量为 0; b) $W(t)$ 非负且随时间 t 单调递增; c) 对 $W(t)$ 进行求导, 则可以得到测试工作量增长速率函数 $w(t)$ 为

$$w(t) = W_{\max} \exp(-\frac{\alpha t}{1 + \psi \exp(-\alpha t)}) \times (\frac{\alpha}{1 + \psi \exp(-\alpha t)} + \frac{\alpha^2 \psi t \exp(-\alpha t)}{(1 + \psi \exp(-\alpha t))^2}) \quad (17)$$

5 考虑测试工作量和故障检测率的 NHPP 模型

综上所述, 对测试工作量函数和故障检测率函数进行了描述, 可以利用式 (12) (14) 和式 (16) 进行组合代入式 (9) 进行求解得到两种测试工作量和故障检测率相结合的 NHPP 模型。

1) LF-IS-TEF 模型

$$m(t) = a \times (1 - \exp(\frac{1}{c \times d} - \frac{\ln(d+1) \times (d+1)}{c \times d^2})) \times \exp(\frac{\ln(d + \exp(c \times W(t))) \times (d+1)}{c \times d^2} - \exp(-c \times W(t)) \times (\frac{1}{c \times d} + \frac{W(t) \times \exp(c \times W(t)) \times (d+1)}{d^2})) \quad (18)$$

其中: $W(t) = W_{\max} \frac{1 - e^{-rt}}{1 + \psi e^{-rt}}$ 。

2) LF-LTCS-TEF 模型

$$m(t) = a \times (1 - \exp(\frac{1}{c \times d} - \frac{\ln(d+1) \times (d+1)}{c \times d^2})) \times \exp(\frac{\ln(d + \exp(c \times W(t))) \times (d+1)}{c \times d^2} - \exp(-c \times W(t)) \times (\frac{1}{c \times d} + \frac{W(t) \times \exp(c \times W(t)) \times (d+1)}{d^2})) \quad (19)$$

其中: $W(t) = W_{\max} (1 - \exp(-\frac{\alpha t}{1 + \psi \exp(-\alpha t)}))$ 。

6 实例验证

本文利用两组公开数据集 Ohba^[5] 和 Wood^[5] 来进行实例验证,这两组数据均包含测试时间(t)、测试工作量(W)和检测到的故障数(m)。通过多种具有代表性的可靠性模型对比分析验证本文所提出的模型。本文选用平均误差平方和(means squared error, MSE)作为模型拟合能力的评判标准, MSE 的值越小,则说明拟合能力越好。

$$MSE = \frac{\sum_{i=1}^n [y_i - \hat{m}(t_i)]^2}{n} \quad (20)$$

其中: n 表示失效数据集中失效样本的数量, y_i 为时刻 t_i 发现的故障数, $\hat{m}(t_i)$ 表示时刻 t_i 模型估算到的故障数。

本文选用最小二乘法(LSE)作为模型参数的估计方法,它具有无偏性。实例的验证过程分为两个步骤:a)应用模型框

架来分别建立适于两组公开的失效数据集的测试工作量函数和故障检测率函数的 NHPP 模型;b)将本文中的结合测试工作量函数和故障检测率函数的 NHPP 模型同其他模型进行比较。

6.1 不同的故障检测率的 NHPP 模型比较

根据 Ohba 和 Wood 这两组公开的数据来和如表 1 所示的五种已有的可靠性模型进行验证和比较,具体计算结果如表 1 和表 2 所示。由表 1 和表 2 可知,其中粗体表示最优的结果,在 Ohba 的数据集中 LF 模型中的 MSE 的值稍逊于 LTCS 模型。在 Wood 的数据集中 LF 模型的结果是最优。综上所述 LF 模型在两组数据中都有较好的拟合效果。

6.2 不同的测试工作量函数的比较

利用 Ohba 和 Wood 两组数据集中的测试时间和 CPU 时间来估计测试工作量模型中的参数值并对比模型的拟合效果。本实例将 Wood 数据集中测试工作量数据缩小 100 倍,这种缩放处理不会对模型拟合性能对比结果产生影响^[5],目的是提高参数的估计质量。具体计算结果如表 3 和表 4 所示。

表 1 六种软件可靠性模型在 Ohba 数据上的计算结果

序号	软件可靠性模型	均值函数 $m(t)$	参数取值	MSE
1	G-O Model(GO) ^[2]	$a(1 - \exp(-rt))$	$a = 760.53, r = 0.0323$	139.82
2	Delay S-Shaped Model(DDS) ^[3]	$a(1 - (1 + rt)\exp(-rt))$	$a = 374.05, r = 0.197$	168.67
3	Inflected S-Shaped Model(ISS) ^[4]	$a \frac{1 - \exp(-rt)}{1 + \psi \exp(-rt)}$	$a = 374.39, r = 0.179, \psi = 3.01$	127.31
4	Yamada Weibull Model(YW) ^[13]	$a(1 - \exp(-r\alpha(1 - \exp(-\beta t^\gamma))))$	$a = 460.81, r = 0.0272, \alpha = 49.63, \beta = 0.00708, \gamma = 1.98$	260.04
5	LTCS Model(LTCS) ^[12]	$a(1 - \exp(\frac{-\alpha t}{1 + A \exp(-\alpha t)}))$	$a = 394.6, A = 1.7658, \alpha = 0.1202$	86.05
6	本文模型(LF)	Equation(13)	$a = 393.90, d = 3.6011, c = 0.03133$	88.72

表 2 六种软件可靠性模型在 Wood 数据上的计算结果

序号	软件可靠性模型	均值函数 $m(t)$	参数取值	MSE
1	G-O Model(GO) ^[2]	$a(1 - \exp(-rt))$	$a = 130.20, r = 0.0832$	11.62
2	Delay S-Shaped Model(DDS) ^[3]	$a(1 - (1 + rt)\exp(-rt))$	$a = 103.98, r = 0.265$	25.26
3	Inflected S-Shaped Model(ISS) ^[4]	$a \frac{1 - \exp(-rt)}{1 + \psi \exp(-rt)}$	$a = 110.83, r = 0.172, \psi = 1.2$	8.98
4	Yamada Weibull Model(YW) ^[13]	$a(1 - \exp(-r\alpha(1 - \exp(-\beta t^\gamma))))$	$a = 135.74, r = 0.000142, \alpha = 11710.75, \beta = 0.0235, \gamma = 1.46$	16.59
5	LTCS Model(LTCS) ^[12]	$a(1 - \exp(\frac{-\alpha t}{1 + A \exp(-\alpha t)}))$	$a = 113.7, A = 0.6843, \alpha = 0.1276$	9.41
6	本文模型(LF)	Equation(13)	$a = 109.59, d = 2.647, c = 0.0222$	8.83

表 3 四种测试工作量模型在 Ohba 数据上的计算结果

序号	测试工作量模型	测试工作量函数 $W(t)$	参数取值	MSE
1	EW-TEF ^[14]	$W_{\max}((1 - \exp(-r \times t^m)))^\theta$	$W_{\max} = 107.1, r = 0.00211, m = 2.283, \theta = 0.56$	1.7
2	LO-TEF ^[15]	$\frac{W_{\max}}{1 + A \times \exp(-\alpha t)}$	$W_{\max} = 100, A = 13.15, \alpha = 0.2866$	3.78
3	IS-TEF ^[11]	$W_{\max} \frac{1 - \exp(-rt)}{1 + \psi \exp(-rt)}$	$W_{\max} = 108.5, r = 0.1981, \psi = 4.554$	1.35
4	LTCS-TEF	Equation(16)	$W_{\max} = 110.9, \alpha = 0.1311, \psi = 2.943$	1.33

表 4 四种测试工作量模型在 Wood 数据上的计算结果

序号	测试工作量模型	测试工作量函数 $W(t)$	参数取值	MSE
1	EW-TEF ^[14]	$W_{\max}((1 - \exp(-r \times t^m)))^\theta$	$W_{\max} = 886, r = 0.1111, m = 0.4619, \theta = 2.795$	0.91
2	LO-TEF ^[15]	$\frac{W_{\max}}{1 + A \times \exp(-\alpha t)}$	$W_{\max} = 54.84, A = 13.03, \alpha = 0.2263$	1.63
3	IS-TEF ^[11]	$W_{\max} \frac{1 - \exp(-rt)}{1 + \psi \exp(-rt)}$	$W_{\max} = 91.58, r = 0.09013, \psi = 3.164$	0.83
4	LTCS-TEF	Equation(16)	$W_{\max} = 101.34, \alpha = 0.05532, \psi = 1.852$	0.83

由表 3 和表 4 可知,IS-TEF 和 LTCS-TEF 两者的拟合结果都是比较好,并且两者的结果都相差不大。由此可说明,这两个模型都是比较适合这两组数据的测试工作量函数。

6.3 考虑测试工作量和故障检测率的 NHPP 模型的比较

根据对测试工作量函数中的参数估计,再利用这两组数据中的测试时间和累积故障数,对文中提出的 NHPP 模型的剩下

的参数进行估计并对模型的拟合能力进行比较。具体计算结果如表 5 和表 6 所示。

表 5 六种测试工作量和故障检测率结合模型在 Ohba 数据上的计算结果

序号	软件可靠性模型	均值函数 $m(t)$	参数取值	MSE
1	IS-TEFM ^[11]	$a(1 - \exp(-bW_{\max} \frac{1 - \exp(-rt)}{1 + \psi \exp(-rt)}))$	$a = 564.2, b = 0.01972$	107.05
2	IS-LO-SRGM ^[16]	$a(1 - \exp(\frac{-\alpha W_{\max} \frac{1 - \exp(-rt)}{1 + \psi \exp(-rt)}}{1 + A \exp(-\alpha W_{\max} \frac{1 - \exp(-rt)}{1 + \psi \exp(-rt)})))$	$a = 401.3, A = 0.9388, \alpha = 0.043$	83.05
3	EWE-TEFM ^[14]	$a(1 - \exp(-bW_{\max}((1 - \exp(-r \times t^m)))^\theta))$	$a = 575.7, b = 0.01916$	113.68
4	LO-TEFM ^[15]	$a(1 - \exp(-b(\frac{W_{\max}}{1 + A \times \exp(-\alpha t)} - \frac{W_{\max}}{1 + A})))$	$a = 395.6, b = 0.04164$	114.05
5	本文模型 (IS-LF-SRGM)	Equation (18)	$a = 381.1, d = 4.89, c = 0.00517$	80.83
6	本文模型 (LTCS-LF-SRGM)	Equation (19)	$a = 381.28, d = 4.89, c = 0.00515$	81.75

注:表中的测试工作量函数相关参数见表 3。

表 6 六种测试工作量和故障检测率结合模型在 Wood 数据上的计算结果

序号	软件可靠性模型	均值函数 $m(t)$	参数取值	MSE
1	IS-TEFM ^[11]	$a(1 - \exp(-bW_{\max} \frac{1 - \exp(-rt)}{1 + \psi \exp(-rt)}))$	$a = 135.35, b = 0.0143$	10.03
2	IS-LO-SRGM ^[16]	$a(1 - \exp(\frac{-\alpha W_{\max} \frac{1 - \exp(-rt)}{1 + \psi \exp(-rt)}}{1 + A \exp(-\alpha W_{\max} \frac{1 - \exp(-rt)}{1 + \psi \exp(-rt)})))$	$a = 135.4, A = 8.612e-08, \alpha = 0.0143$	10.03
3	EWE-TEFM ^[14]	$a(1 - \exp(-bW_{\max}((1 - \exp(-r \times t^m)))^\theta))$	$a = 107.1, b = 0.0142$	13.31
4	LO-TEFM ^[15]	$a(1 - \exp(-b(\frac{W_{\max}}{1 + A \times \exp(-\alpha t)} - \frac{W_{\max}}{1 + A})))$	$a = 112.1, b = 0.02409$	21.44
5	本文模型 (IS-LF-SRGM)	Equation (18)	$a = 141.47, d = 44.61, c = 0.9299$	7.91
6	本文模型 (LTCS-LF-SRGM)	Equation (19)	$a = 141.07, d = 44.36, c = 0.9485$	8.54

注:表中的测试工作量函数相关参数见表 4。

根据表 5 和表 6 的对比结果可知,IS-LF-SRGM 模型的拟合能力在两组数据中都是最优的,LTCS-LF-SRGM 模型的拟合能力稍差。

7 结束语

本文提出的 LF 模型重点描述了故障检测率随时间变化的情况,它呈现的是一种先增后减的趋势,该模型更加符合实际情况。本文中的 IS-LF-SRGM 和 LTCS-LF-SRGM 模型是将测试工作量这个因素引入到 LF 模型中。通过实验数据证明,本文中的模型有较好的拟合效果和实际意义。在今后的工作中,将考虑在模型中加入其他影响软件可靠性的因素,例如测试覆盖率等,来进一步改进提高模型的适应能力。

参考文献:

- [1] IEEE Std 1633-2008, IEEE Recommended Practice on software reliability[S]. 2008.
- [2] Musa J D. Software reliability engineering[M]. New York: McGraw-Hill, 1999.
- [3] Pham H. System software reliability[M]. Heidelberg: Springer, 2006.
- [4] Xie M. Software reliability modeling[M]. Singapore: World Scientific Publishing Company, 1991.
- [5] Huanh C Y, Kuo S Y, Lyu M R. An assessment of testing-effort dependent software reliability growth models[J]. IEEE Trans on Reliability, 2007, 56(2): 198-211.
- [6] Kuo S Y, Huang C Y, Lyu M R. Framework for modeling software reliability, using various testing-efforts and fault-detection rates[J]. IEEE Trans on Reliability, 2001, 50(3): 310-320.
- [7] Ahmad N, Khan M G M, Rafi L S. A study of testing effort dependent

inflection S-shaped software reliability growth models with imperfect debugging[J]. International Journal of Quality & Reliability Management, 2010, 27(1): 89-110.

- [8] 吴开贵, 张辉, 尹鲁燕, 等. 故障察觉率可变的软件可靠性增长模型[J]. 计算机系统应用, 2011, 20(9): 103-106.
- [9] Hsu C J, Huang C Y, Chang J R. Enhancing software reliability modeling and prediction through the introduction of time variable fault reduction factor[J]. Applied Mathematical Modeling, 2011, 35(1): 506-521.
- [10] 刘宏伟, 杨孝宗, 曲峰, 等. 一个基于响铃形故障检测率函数的软件可靠性增长模型[J]. 计算机学报, 2005, 28(5): 908-913.
- [11] 李秋英, 李海峰, 陆民燕, 等. 基于 S 型测试工作量函数的软件可靠性增长模型[J]. 北京航空航天大学学报, 2011, 37(2): 149-155.
- [12] 李海峰, 李秋英, 陆民燕. 基于 Logistic 测试覆盖率函数的软件可靠性建模研究[J]. 计算机研究与发展, 2011, 48(2): 232-240.
- [13] Li Qiuying, Li Haifeng, Lu Minyan. Incorporating S-shaped testing effort functions into NHPP software reliability model with imperfect debugging[J]. Journal of Systems Engineering and Electronics, 2015, 26(1): 190-207.
- [14] Huang C Y, Kuo S Y. Analysis of incorporating logistic testing-effort function into software reliability modeling[J]. IEEE Trans on Reliability, 2002, 51(3): 261-270.
- [15] Ahmad N, Bokhari M U, Quadri S M K, et al. The exponentiated weibull software reliability growth model with various testing-efforts and optimal release policy[J]. International Journal of Quality & Reliability Management, 2008, 25(2): 211-235.
- [16] 李海峰, 王栓奇, 刘畅, 等. 考虑测试工作量与测试覆盖率的软件可靠性模型[J]. 软件学报, 2013, 24(4): 749-760.