浙江理工大学学报(自然科学版),第35卷,第1期,2016年1月
Journal of Zhejiang Sci-Tech University (Natural Sciences)
Vol. 35, No. 1, Jan. 2016

DOI:10.3969/j.issn.1673-3851.2016.01.013

考虑缺陷关联模型的软件优化测试策略

姚　弋[1],谢晓鸣[2],刘书涵[2],张　娜[2],俞成海[2]
(1.浙江省消防总队,杭州 310014;2.浙江理工大学信息学院,杭州 310018)

摘　要: 软件测试中,缺陷关联是一种比较普遍的现象。已有研究表明,充分利用缺陷之间的关联信息有助于提高软件测试效率,同时,从缺陷自身角度分析,为了综合权衡多个影响测试决策的因素(如缺陷等级、缺陷可检测率和缺陷关联关系)。提出一种基于缺陷关联的受控马尔科夫链测试模型,将测试过程转换成基于多目标权值的路径优化问题,综合权衡缺陷可检测率、关联系数和测试回扣;并运用Prim算法构造最小生成树以构造基于多目标权值的软件优化测试策略。在资源约束下,该方法利用缺陷关联引导测试决策的选取,优先检测关联紧密、可检测率大、回扣多的关联缺陷。通过仿真实验,证明了该方法的有效性。

关键词: 软件测试;受控马尔科夫链;关联缺陷;多目标权值;Prim算法
中图分类号: TP311　**文献标志码:** A　**文章编号:** 1673-3851(2016)01-0078-06　**引用页码:** 010601

0　引　言

由于软件开发过程仍严重依赖开发人员的技术和经验,缺乏客观统一的评价标准,导致软件失效的现象频发,软件可靠性[1]问题已成为制约软件产业发展的瓶颈之一。作为软件质量保证的重要手段,软件测试已成为学术界和工业界研究的重点。目前,部分研究者通过受控马尔可夫链(controlled markov chain, CMC)理论构建测试模型[2],在资源约束下,希望以最小的成本检测并剔除尽可能多的缺陷。但不同种类的软件缺陷所引发的失效程度不同,因而缺陷被剔除后所产生的测试价值(回扣值)也不同,这就导致测试目标不仅仅是检测更多的缺陷数量,同时也应该兼顾测试价值。发现一个致命的软件缺陷显然比检测到一些轻微的缺陷能产生更多的测试价值。此外,在实际测试过程中,缺陷之间往往相互关联。尽管软件企业力求以高内聚低耦合[3]的标准去开发产品,但对多数程序而言,其所选择语句的运行或多或少都要受到程序中其他已执行语句的影响,这表明缺陷之间存在着关联关系。研究表明,集成相同或者相似功能的测试用例通常能够检测出包含该特性的缺陷[4]。为此,如何有效地运用缺陷关联性,并结合已有的测试模型设计基于多目标的软件测试策略成为本文研究的重点。

本文在前期研究成果上[2],提出了一种引入缺陷关联度的CMC测试模型,将缺陷关联系数、缺陷可检测率和回扣三者作为权值,以支持基于多目标权值的决策选取标准。利用缺陷之间的关联信息,以软件缺陷作为顶点,关联关系当作路径,将测试过程转换成求解带权路径优化问题;运用Prim算法构造最小生成树以获得基于多目标的软件缺陷优化测试策略。该方法利用关联缺陷引导测试决策的选取,优先检测关联紧密、可检测率大、回扣多的关联缺陷集。

1　相关工作

本文研究的内容与关联缺陷和基于受控马尔科夫链测试模型的研究密切相关。

为形式化描述测试过程中软件状态的改变,降低建模复杂度,部分研究者对受控马尔科夫链测试

收稿日期:2014-11-27
基金项目:国家自然科学基金项目(61379036,61502430);浙江省自然科学基金项目(Y13F020175,LY12F02041);浙江省新苗计划项目(2014R406073);浙江理工大学521人才培养计划
作者简介:姚　弋(1975-),男,浙江诸暨人,高级工程师,主要从事数据库及软件测试技术方面的研究。
通信作者:张　娜,E-mail:zhangna@zstu.edu.cn

模型进行了一些特殊化的处理,主要包括[5-7]:a)假定被测软件包含的缺陷数一定,b)假定不同的软件缺陷被检测到的概率相等,c)假定检测到一个缺陷就立即剔除,d)假定剔除缺陷的决策不消耗测试资源(即代价),e)假定剔除不同缺陷产生的回扣相等。为了缩小测试模型与实际运行时的环境差异,Hu等[8]完善了模型的假设条件,包括考虑了不同种类缺陷具有不同的可检测率;启用批量剔除策略等,提高了模型的实用性。包晓安等[2]对资源受限环境下的测试过程进行了探索,提出了一种资源受约束的CMC测试模型,并以此构造软件自适应测试策略。虽然这些测试技术都经过了理论分析和实践验证,较好的提升了软件测试效率,但他们大都存在一个共同的缺陷:即忽略了软件缺陷之间的关联关系。

在实际测试过程中,测试人员通常会发现缺陷之间存在某种关联,即失效关联,而这种关联关系可以影响相关缺陷的检测能力,为测试人员提供某种启发式的方法,从而对测试结果产生影响。李鹏等[9]基于面向对象的设计模型,分析了软件缺陷的传播过程。针对关联缺陷的检测,景涛等[10]从软件缺陷角度分析失效关联,提出一种基于缺陷放回的测试方法。这些文献大多从缺陷关联的现象进行分析,评估关联失效对软件可靠性造成的影响,认为关联缺陷是造成软件失效的根源之一。但已有研究也表明[11],在单元测试中发现的多个程序错误很大程度上来自于同一函数或者模块,受到相同程序结构的影响,所以存在的错误往往具有相似性与关联性。因此,如何利用缺陷关联信息引导软件测试过程,权衡多个影响测试决策选取的因素,设计基于多目标的优化测试策略成为本文研究的重点。

2　基于缺陷关联的软件测试模型

2.1　关联缺陷定义

针对关联缺陷的定义基于以下假设:

a)对于一个软件系统,定义 $S_t = S_t(n)$ 表示 t 时刻软件系统剩余缺陷数,$S_t(0)$ 表示系统的终止态,表明系统中剩余缺陷为0,测试过程停止;

b) 缺陷集合 $Defects = \{d_1, d_2, \cdots, d_n\}$;$n > 0$,$n$ 为总缺陷数;测试用例库 $TCase = \{tc_1, tc_2, \cdots, tc_m\}$,$m > 0$,$m$ 为测试用例总数;

c) 缺陷被修复后不引入新的缺陷;

定义1　将 $DDP(d_i)$ 表示为 t 时刻下测试用例 tc_m 检测到缺陷 d_i 的概率,其中 DDP 表示缺陷检测率(defect detected probability,DDP),$tc_m \in TCase$,$d_i \in Defects$。

定义2　对于同一个 tc_m 而言,由于 d_i 被检测到而使得 d_j 被检测到的概率变为 $DDP_1(d_j)$,则称 d_i 关联 d_j,记为 $d_i \& d_j = \{(d_i, d_j) \mid r_{ij}\}$,其中关联系数 r_{ij} 表示缺陷之间的关联程度。

2.1　基于Markov决策的测试模型构建

原有的测试模型通过缺陷检测率来实现软件状态的改变,没有考虑每个缺陷的重要性和关联性。本文引入缺陷关联系数,并结合缺陷检测率和回扣组成多目标权值作为软件状态转移标准。针对软件测试过程,本文的模型[12]基于以下假设:

a) 测试开始时($t = 0$ 时刻),软件系统包含 n 个缺陷。

b) 一般地,将缺陷分成"已剔除"、"未检测到"、"已检测到但未移除"这三种状态,分别表示为 S_0,S_1,S_2。

$$Y_t^k = \begin{cases} S_0, t \text{ 时刻第 } k \text{ 个缺陷已移除} \\ S_1, t \text{ 时刻第 } k \text{ 个缺陷未被检测到} \\ S_2, t \text{ 时刻第 } k \text{ 个缺陷已经检测到但未被移除} \end{cases} \tag{1}$$

其中 $k = 1, 2, \cdots, n$.

c) 软件初始状态为 $S_t = (Y_t^k, x_t)$,x_t 表示 t 时刻系统可用测试资源,当 x_t 小于执行任意决策所需的代价或所有缺陷都已被移除时,相应的软件状态更新为吸收态 $S_t = (0,0)$,表明测试活动停止。

d) 在每个时刻 t,总有 $m+1$ 个可选决策 $A = \{1, 2\cdots m, rev\}$,其中前 m 个决策用于检测缺陷,每个决策消耗资源 $C_{\xi_t}(A_t)$,rev 用于剔除缺陷,其执行代价为 C_1,每当检测到 d 个缺陷时候就执行一次批量剔除。

e) 元素 r_{ij} 代表缺陷 i 与缺陷 j 关联且具有值为 r_{ij} 的关联性。R_{ij} 组成 n 阶缺陷关联矩阵 Ω:

$$\Omega = \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ r_{21} & r_{22} & \cdots & r_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ r_{n1} & r_{n2} & \cdots & r_{m} \end{bmatrix},$$

$$\begin{cases} r_{ij} = 1, i = j, \text{表示缺陷的自关联} \\ r_{ij} = 0, \text{表示缺陷之间不存在关联} \\ r_{ij} = \alpha_{ij} i \neq j, \text{表示存在系数为 } \alpha_{ij} \\ \qquad \text{的关联性且 } \alpha_{ij} \in (0,1) \end{cases} \tag{2}$$

f) 缺陷检测率用矩阵 Θ 表示:

$$\Theta = \begin{bmatrix} \rho_1^{(1)} & \rho_1^{(2)} & \cdots & \rho_1^{(n)} \\ \rho_2^{(1)} & \rho_2^{(2)} & \cdots & \rho_2^{(n)} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_m^{(1)} & \rho_m^{(2)} & \cdots & \rho_m^{(n)} \end{bmatrix} \tag{3}$$

其中:$k = 1, 2, \cdots, n$,向量 $\Theta = [\rho_{k_1}^{(1)}, \rho_{k_2}^{(2)} \cdots \rho_{k_m}^{(n)}]$ 为矩

阵 Θ 的最大元素列元素所组成。

g) 在决策 A_t 下，某个缺陷被检测并剔除后获得一定的测试价值，即产生回扣 δ_x。缺陷的性质、成因以及其被检测和修复的成本决定了其测试价值，对于那些包含在交互接口、底层框架、系统核心业务之中的缺陷，尤其具有较高的测试价值，因而也能产生较大的测试回扣。为了表示不同重要性等级的缺陷所产生的测试价值，将每个缺陷的期望回扣用向量 $\delta=[\delta_1,\delta_2,\cdots,\delta_n]$ 表示。

根据构建的模型设计价值函数，$v=v_{A_t}(\rho_i^j,r_{ij},\delta_i)$ 用于计算关联缺陷产生的测试价值。其中参数 $\rho_i^j(1\leqslant i\leqslant n,1\leqslant j\leqslant m)$ 为缺陷被检测到的概率，r_{ij} 为关联系数，δ_i 表示回扣。令 τ 为软件首次到达吸收态的时间，则总的测试资源满足以下等式：

$$x_0=\sum_{t=0}^{\tau}[C_{S_t}(A_t)-v_{A_t}(\rho_j^i,r_{ij},\delta_i)]+x_\tau+\left(\frac{n}{d}+1\right)C_1 \tag{4}$$

其中：x_0 和 x_r 分别表示初始资源跟剩余资源，$C_{S_t}(A_t)$ 表示每次检测所需的资源，$(n/d+1)C_1$ 表示执行批量剔除消耗的资源。则有：

$$J_w(x_0)=\sum_{t=0}^{t}v_{A_t}(\rho_j^i,r_{ij},\delta_t) \tag{5}$$

其中：$J_\omega(x_0)$ 表示到 t 时刻为止产生的回扣总量。在给定初始资源的条件下，本文的研究问题就转化为设计一个基于缺陷关联模型的测试剖面，以使得 $v=v_{A_t}(\rho_i^j,r_{ij},\delta_i)$ 函数取得最大值。

3 多目标优化测试策略设计

由于关联缺陷的存在，同时为了获取价值函数 $v=v_{A_t}(\rho_i^j,r_{ij},\delta_i)$ 的极值，将测试过程看作一张无向连通图：以被测缺陷作为顶点，关联系数作为路径，缺陷可检测率、回扣和关联系数三者作为基于多目标的路径权值。通过 Prim 算法对具有最大权值的连通分量构造最小生成树，以最少的成本连通具有关联关系的缺陷集。借助构造最小生成树，获取缺陷顶点被访问的顺序，从而得到基于多目标权值的决策选取序列。该方法使得测试决策在综合权衡多目标权值的基础上，优先检测权值高的关联缺陷，优化缺陷剔除回扣。

将待测缺陷作为图中顶点，则 $V=\{1,2,3,\cdots,n\}$，令 W_{ij} 为边的权值，用矩阵表示如下：

$$W=\begin{bmatrix}1 & w_{12} & \cdots & w_{1n}\\ w_{21} & 1 & \cdots & w_{2n}\\ \vdots & \vdots & \ddots & \vdots\\ w_{n1} & w_{n2} & \cdots & 1\end{bmatrix},$$

其中：$w_{ij}=\delta_i*\delta_j*r_{ij}*\rho_{k_1}^i*\rho_{k_2}^j$，$(1\leqslant i\leqslant n,1\leqslant j\leqslant n,1\leqslant k_1\leqslant m,1\leqslant k_2\leqslant m)$

即用关联缺陷系数、缺陷回扣、缺陷可检测率的乘积表示缺陷 i 与 j 之间的权值。为了获得基于权值的测试决策，需要在 W 中寻找包含最大权值的连通分量，并运用 Prim 算法来构造最小生成树以获得缺陷的遍历序列。

基于 Prim 算法的测试决策生成：

a) Initial $S,v\in U,u\in V-U$ //Te 为 S 中边的集合，V 为 S 中顶点的集合，(v,u) 表示一条具有最大权值的边；

b) $S=$ DFS_MaxWeight(w_{ij})// 搜索具有最大权值的连通分量；

c) $K=$ InitVex(S,U)；// 从第 U 个顶点出发构造 S 的最小生成树 T，记录 T 的各个顶点；

d) Edge_array[S. vexnum]// 设置辅助数组，记录 S 中具有最大权值的边；

e) Init_Edge()；// 初始化辅助数组；

f) 初始化 $U=\{u_0\}$；

g) Repeat until $U=V$；

h) $k=$ MaxWeight(Edge_array)；// 求出 T 的下一个结点第 k 顶点；

i) //closedge[k]. highcost = Max{closedge[v_i]. highcost | closedge[v_i]. highcost >0, $v_i\in V-U$}；

j) addTe(v_i,u_i)；// 在 S 寻找一条权值最大的边并入集合 Te；

k) addU(u_i)；// 将 u_0 并入 U，直到 $U=V$；

l) Output()// 依次输出最小生成树的边 $G=\{(v_0,u_0),(v_1,u_1)\cdots(v_i,u_i)\}$；

由 Prim 算法的测试决策生成可得，集合 G 中的序列记录了构造最小生成树的顶点顺序。当测试过程中目标函数 $v=v_{A_t}(\rho_i^j,r_{ij},\delta_i)$ 的决策 A_t 选取满足集合 G 中的序列时，$v=v_{A_t}(\rho_i^j,r_{ij},\delta_i)$ 能产生最大的回扣，那么相应的决策 A_t 即为本文的优化测试策略。该方法综合考虑了测试价值、缺陷可检测率和缺陷关联性这三方面，通过权值大小作为测试决策选取的标准，从而优先检测回扣高、可检测率大、关联性强的缺陷集。

4 仿真验证

本文选取随机测试和参数已知的自适应测试作为比较标准，来验证测试模型和策略的有效性。在随机测试中每个决策被选取的概率相等，缺陷检测顺序随机选择。为了减少自适应测试中参数估计的准

确性,避免对测试结果的干扰,本文用 0.8 作为实际参数估计的结果进行拟合。

假设测试开始时,令 $n=7$,$d=2$,$A_t=\{1,2\cdots11\}$,$\delta_i=[10,15,40,25,20,18]$,$W_{\xi_t}(1)=W_{\xi_t}(2)=\cdots=W_{\xi_t}(10)=15$;缺陷检测率矩阵与缺陷关联矩阵分别为:

$$\Theta=\begin{bmatrix}0.05&0.17&0.2&0.15&0.24&0.45&0.27\\0.05&0.15&0.18&0.10&0.33&0.38&0.25\\0.04&0.19&0.22&0.14&0.36&0.36&0.18\\0.06&0.18&0.24&0.15&0.30&0.37&0.25\\0.05&0.20&0.19&0.09&0.18&0.43&0.30\\0.06&0.15&0.23&0.12&0.27&0.33&0.26\\0.03&0.17&0.18&0.16&0.35&0.22&0.18\\0.06&0.12&0.13&0.15&0.28&0.40&0.31\\0.07&0.19&0.25&0.13&0.25&0.35&0.25\\0.08&0.18&0.21&0.15&0.31&0.28&0.24\end{bmatrix},$$

$$\Omega=\begin{bmatrix}1.00&0.00&0.05&0.04&0.06&0.08&0.00\\0.00&1.00&0.00&0.00&0.00&0.00&0.00\\0.03&0.00&1.00&0.03&0.03&0.00&0.00\\0.05&0.00&0.04&1.00&0.00&0.04&0.00\\0.08&0.00&0.05&0.00&1.00&0.03&0.00\\0.10&0.00&0.00&0.05&0.02&1.00&0.00\\0.00&0.00&0.00&0.00&0.00&0.00&1.00\end{bmatrix}.$$

为了简化计算复杂度,将关联系数转化为整数带入公式,计算得权值矩阵为:

$$W=\begin{bmatrix}1.000&0.000&108.000&40.960&165.888&207.360&0.000\\0.000&1.000&0.000&0.000&0.000&0.000&0.000\\64.800&0.000&1.000&64.800&174.960&0.000&0.000\\51.200&0.000&86.400&1.000&0.000&103.680&0.000\\221.184&0.000&291.600&0.000&1.000&209.952&0.000\\259.200&0.000&0.000&138.240&139.968&1.000&0.000\\0.000&0.000&0.000&0.000&0.000&0.000&1.000\end{bmatrix}.$$

所构造的最小生成树顶点集合为:$G=\{3,5,1,6,4\}$,根据顶点集合G可得检测顺序为$D_i=\{5,3,4,6,1,2,7\}$,那么根据转移概率矩阵可知,决策选择顺序为:$A_t=\{9,3,10,1,7,5,8\}$。

根据以上决策并结合给定的相关参数,将基于多目标的优化测试策略(OT) 分别与随机测试(RT) 和自适应测试(AT) 相比较。为了反映软件状态在测试过程中的变化,从多次测试结果集中随机选取一条软件状态转移路径,并比较它们在 3 种测试策略下的代价,如表 1 所示。从表 1 中数据可知,优化测试策略比随机测试和自适应测试分别少花费约 30.51% 和 20.65% 的代价。为了更形象地反映仿真结果,将表 1 数据绘制成图 1,其中横坐标表示每个决策下的软件状态,纵坐标表示转移代价。

表 1　3 种测试策略下软件状态转移代价的比较

测试策略	软件状态											
	(10,7)	(9,7)	(8,7)	(7,6)	(6,6)	(5,6)	(4,5)	(3,5)	(2,4)	(1,3)	(0,2)	总代价
OT	36.54	31.85	27.32	27.27	22.17	27.63	22.31	16.97	5.59	12.26	0.00	229.91
AT	62.74	56.91	51.03	28.66	23.58	18.65	18.83	13.6	13.53	2.22	0.00	289.75
RT	69.43	62.78	56.12	34.13	27.99	22.16	22.04	15.74	16.06	4.39	0.00	330.84

表 2　剔除的缺陷数和回扣值比较

i	基于多目标的优化测试(OT)		自适应测试(AT)		随机测试(RT)	
	剔除缺陷数	产生回扣	剔除缺陷数	产生回扣	剔除缺陷数	产生回扣
1	4.26	112.56	3.81	95.00	2.85	75.85
2	4.24	112.19	3.77	93.27	2.70	71.32
3	4.14	109.96	3.86	98.16	3.06	81.00
4	4.32	113.50	3.87	96.16	2.98	79.46
5	4.14	109.52	3.85	95.90	2.80	75.37
均值	4.22	111.546	3.832	95.968	2.878	76.60

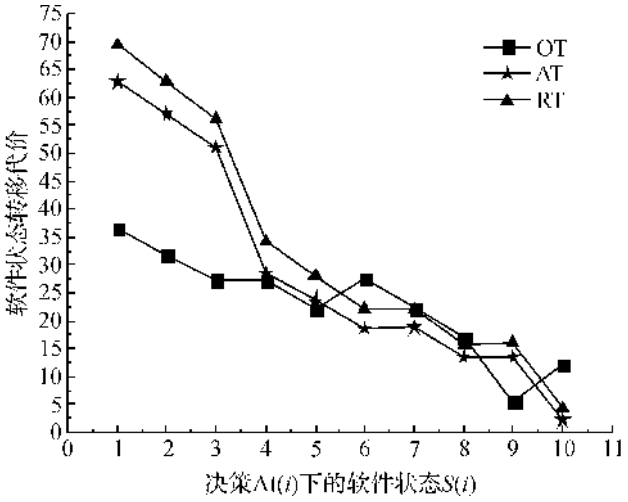

图1 3种测试策略下软件状态转移代价的比较

在初始资源相同的条件下，由于假设执行决策所消耗的资源相同，为了显示不同方法在检测缺陷数和测试价值的差异，将问题转化为在有限测试次数下比较3种方法剔除的缺陷数和产生的回扣值。表2显示了在测试步数为30时，3种测试策略下的剔除缺陷数和回扣值。从表2中可知，OT比RT和AT分别多检测46.63%和10.13%的缺陷数，同时，OT比RT和AT分别多产生45.62%和16.23%的回扣。

相比于缺陷独立的假设，基于多目标的优化测试略充分利用缺陷之间的关联信息，提高相关缺陷的检测率，并根据多目标权值选取测试决策，因而能够产生更多的回扣。相比与其他两种方法，本文的测试方法在软件状态转移代价、缺陷剔除数和产生回扣上具有一定的优势。

5 结 语

在前期工作基础上，本文利用关联缺陷性质改进受控马尔科夫链测试模型，将测试过程转换成基于多目标的带权路径优化问题，并运用Prim算法构造最小生成树，以获取基于缺陷回扣、缺陷检测率、关联系数的决策选取序列。通过与其他两种方法进行比较，仿真实验结果表明，本文所提的测试策略是可行和有效的。

参考文献：

[1] KAPUR P K, PHAN H, ANAND S, et al. A unified approach for developing software reliability growth models in the presence of imperfect debugging and error generation [J]. IEEE Trans on Reliability, 2011, 60(1): 331-340.

[2] 包晓安，姚澜，张晓文，等. 基于受控马尔科夫链的软件缺陷优化测试策略[J]. 计算机科学，2012，39(5)：117-119.

[3] 张莉，钱冠群，李琳. 基于变更传播仿真的软件稳定性分析[J]. 计算机学报，2010，33(3)：440-451.

[4] ZHAO D, UPADHYAYA S. Dynamically partitioned test scheduling with adaptive TAM configuration for power-constrained SoC testing [J]. IEEE Transaction on Computer-aided Design of Integrated Circuits and Systems, 2005, 24(6): 956-965.

[5] CAI K Y. Optimal software testing and adaptive software testing in the context of software cybernetics [J]. Information and Software Technology, 2002, 44(02): 841-855.

[6] WEYNS K, RUNESON P. Sensitivity of software system reliability to usage profile changes [C]// Proceedings of the 2007 ACM symposium on Applied computing. ACM, 2007: 1440-1444.

[7] CAI K Y, DONG Z, LIU K, et al. A mathematical modeling framework for software reliability testing[J]. International Journal of General Systems, 2006, 36(4): 399-463.

[8] HU H, JIANG C H, CAI KY. Adaptive software testing in the context of an improved controlled Markov chain model[C]//Computer Software and Applications, 2008. COMPSAC'08. 32nd Annual IEEE International. IEEE, 2008: 853-858.

[9] 李鹏，赵逢禹. 软件缺陷关联分析与缺陷排除研究[J]. 计算机科学，2013，40(10)：159-161.

[10] 景涛，江昌海，胡得斌，等. 软件关联缺陷的一种检测方法[J]. 软件学报，2005，16(1)：17-18.

[11] 张云乾，郑征，季晓慧，等. 基于马尔可夫模型的软件错误定位方法[J]. 计算机学报，2013，36(2)：445-456.

[12] 包晓安，谢晓鸣，张娜，等. 基于缺陷关联度的Markov模型软件优化测试策略[J]. 软件学报，2015，26(1)：14-25.

Software Optimization Testing Strategy Considering Defect Correlation Model

YAO Yi[1], *XIE Xiaoming*[2], *LIU Shuhan*[2], *ZHANG Na*[2], *YU Chenghai*[2]

(1. Fire Corps of Zhejiang Province, Hangzhou 310014, China; 2. School of Information Science and Technology, Zhejiang Sci-Tech University, Hangzhou 310018, China)

Abstract: In software testing, defect correlation is a relatively common phenomenon. Studies have shown that making the best of associated information between defects is beneficial to improving software testing efficiency. Meanwhile, from the perspective of defects themselves, a controlled Markov chain model based-on the defect correlation is proposed in order to synthetically balance multiple factors influencing the decision-making (such as defect severity level, defect detecting rate and the relationships between correlated detects). The testing is converted to route optimization problem based on multi-objective weight. Besides, defect detection rate, correlation coefficient and testing rebates are measured overall. Prim algorithm is applied to construct minimum spanning tree so as to construct software optimization testing strategy based on multi-objective weight. Under resource constraint, this method utilizes defect correlation to guide selection of testing strategy, and preferentially detect correlated defects with close correlation, large detection rate and many rebates. The simulation experiment proves effectiveness of the method.

Key words: software testing; controlled markov chain; correlated defects; multi-objective weight; Prim algorithm

(责任编辑: 陈和榜)