多维标度法选择回归测试子集

王晓华[1,2,3] 张 涛[1] 尚景亮[1,3] 王金波[1]
(中国科学院光电研究院 北京 100190)[1] (中国科学院空间科学与应用研究中心 北京 100190)[2]
(中国科学院研究生院 北京 100039)[3]

摘 要 在软件改动较大且资源限制严格时,回归测试目前的方法难以满足实际需要。因此,提出利用多维标度法选择回归测试子集的方法。该方法使用测试执行剖面表示测试集,通过计算剖面数据,将测试集合按照测试效果可视化分类,综合考虑了软件变更及测试用例的典型性,适用于大规模软件更改较多时获取有代表性的测试子集。实验表明,多维标度法能够全面反映原测试集对变更的测试程度,从而使得回归子集的获取及测试实施更有针对性,能够满足限制严格的回归测试要求。

关键词 回归测试,多维标度法,覆盖,执行剖面,选择
中图法分类号 TP311.5 **文献标识码** A

Selecting Regression Test Subset by Multidimensional Scaling

WANG Xiao-hua[1,2,3] ZHANG Tao[1] SHANG Jing-liang[1,3] WANG Jin-bo[1]
(Academy of Opto-Electronics, Chinese Academy of Sciences, Beijing 100190, China)[1]
(Center for Space Science and Applied Research, Chinese Academy of Sciences, Beijing 100190, China)[2]
(Graduate University, Chinese Academy of Sciences, Beijing 100039, China)[3]

Abstract To resolve the problem that there are not appropriate techniques for regression testing, when software changes much and testing resources aren't enough, this paper proposed a method to select regression subset by multidimensional scaling. According to the efficacy of the test suite testing, this method classified the test suite visually by computing its execution profiles data which denote the suite. For the method took both software changing and test cases representation into consideration, it was adaptable for obtaining typical regression test subset of big size software which changed much. The experiment indicates that this method can reflect the efficacy of the original test suite testing change and get classical test subset, so it can be satisfied with the strict requirements of regression testing and guiding practical testing.

Keywords Regression test, Multidimensional scaling, Covering, Execution profile, Selection

软件更改就可能引入缺陷,需要进行回归测试,以验证更改部分是否正确及更改是否影响了原有功能。回归测试有全部重测和选择性测试[1]两种策略,其中全部重测代价极大,通常只有时间等资源充分时才选择该策略。

选择性测试策略的方法有随机选择、基于覆盖的选择、基于切片的选择、基于执行剖面的聚类选择等。随机选择是从原测试集中随机抽取或由测试人员根据经验选取测试用例,具有较大的不确定性,难以保证回归质量。基于覆盖的选择[2,3]以测试用例覆盖更改部分的多少为依据选择测试用例,首先选取覆盖更改部分最多的测试用例,接着选择覆盖余下更改最多的测试用例。依次类推,直至覆盖所有更改或达到选择目标为止。基于切片的选择[4,5]本质上也属于覆盖选择,它扩展了需要覆盖的软件更改部分,将所有覆盖更改部分的测试用例的输出作为切片准则来获得各自的相关切片,并将这些程序元素也作为覆盖依据选取测试用例。实践中发现,测试人员往往能通过很少的测试用例就可达到较大的覆盖率,这说明基于覆盖的方法虽然测试了更改部分,但没有考虑测试用例在原测试集中的典型性,无法代表原测试集。特别在软件规模大、更改多,或原测试集数目大时,获得所需测试用例输出语句的相关切片本身就是非常耗时的工作,在资源有限的情况下往往难以有效实施。基于执行剖面的聚类选择[2,3,6,7,9]方法虽然考虑了用例执行的数据信息,能够从相似的用例集中挑选有代表性的测试用例,但测试人员难以直接观察用例测试效果之间的关系。测试是个人经验、智力参与度很高的工作,测试人员如果能够通过观测各用例测试效果的相对分布而参与其中,对回归测试更具指导意义。

基于此,本文提出利用多维标度法选择回归测试集的方法。该方法综合考虑软件变更及测试用例对测试集合的代表性,使得挑选出的测试子集具有很好的典型性及较高的缺陷发现率,能够反映软件质量。

到稿日期:2009-12-17 返修日期:2010-02-25
王晓华(1979—),女,博士生,主要研究方向为软件工程、软件测试,E-mail:saloty@hotmail.com;张 涛 男,研究员,主要研究方向为软件可靠性、软件测试;尚景亮 男,硕士生,主要研究方向为软件测试工具;王金波 男,博士,主要研究方向为软件可靠性、软件测试。

1 研究背景

回归测试需要从原测试集中选择适用的子集检验更改后的程序，如果必要，还需产生新的测试集，验证更改后软件的新增部分。本文的研究内容着重从原测试集中选择适用的子集，也就是令 P 为被测程序，T 为它的原测试集，若 P' 为更改后的程序，则需要找到 $T' \subseteq T$，用 T' 执行 P'，以确认程序 P' 在 T' 的正确性。

回归测试的全部重测策略意味着 $T'=T$。当软件规模大或用例数量多时极耗资源，有时甚至是无法完成的。在航天工程的测试实践中，一个测试用例就可能需运行 5h，在严格的时限下回归重测所有用例是不可能的。

选择性测试策略的重点在于选择合适的 $T' \subset T$，使得 T' 满足变更测试需求 R。令 $R=(r_1,\cdots,r_i,\cdots,r_m)$，$r_i$ 是需要被测试的变更需求条目，可表示变更的函数、语句、覆盖变更的测试用例的输出语句切片，也可表示变更的功能条目等。原测试集合 T 可由一组子集 $T_1,T_2,\cdots,T_n$ 表示，其中可能存在一组测试集合 T_q 对变更测试需求 R 没有测试贡献，在选择回归测试子集时会首先排除这一子集。T 中其余任一子集 T_i 都与部分变更测试需求 $(r_i,\cdots,r_q)$ 相对应，即属于同一子集 T_i 的任一测试用例 t_j 对变更需求的测试程度都类似。回归测试子集的选择就是从 T 中找到代表性的子集 T' 满足所有的变更测试需求 r_i，T' 称为回归命中集合。可以看出，T' 必须包含每个 T_i 中的至少一个测试用例，以达到与原测试集 T 的测试效果等效的目的。

本文使用测试执行剖面表示原测试集 T，利用多维标度法计算执行剖面数据。将 T 按照对变更测试需求 R 的不同测试效果划分为若干子集，以便得到回归命中集合 T'。

2 多维标度选择原理

2.1 测试执行剖面

测试用例的设计通常是基于需求的，从执行输出判断失效是否发生。一次测试能够发现软件缺陷满足 3 个条件：测试执行了包含缺陷的语句；该执行导致软件产生了错误的状态；错误的状态被传递到输出，表现为软件的一次失效[10]。每个测试用例实际执行后都会产生对应的执行剖面。

定义 1(执行剖面) 测试用例执行后，会引起一系列的程序内部事件，如语句、分支的执行，函数的调用等。以程序元素的执行状况表现测试用例的运行轨迹，就称为测试用例的执行剖面。

测试用例的执行剖面展现了程序在该用例下的行为特征。该行为由测试输入引发，由程序结构决定。软件无论功能变更，还是结构变更都会反映到软件行为的变化上。因此，测试执行剖面不仅包含了软件功能信息，也包含了程序结构信息，体现了各用例的测试目的及对软件结构的测试情况。测试集的执行剖面是该测试集引发的软件行为特征集，同样反映了该测试集的测试目的及其对软件结构的测试程度，能够全面展现该测试集的测试效果。各测试执行剖面之间的差异，则显示了不同用例对软件测试范围之间的差别。因而以执行剖面代表测试集 T，通过执行剖面之间的关系划分 T 为子集 $T_1,\cdots,T_n$，能够综合反映对变更测试需求 R 的测试程度，解决了基于覆盖的回归测试选择方法难以在关注结构变更需求的同时考虑各用例功能测试目的的问题。

2.2 多维标度法

多维标度法(multidimensional scaling，MDS)是一种在低维空间展示相似性数据结构的多元统计分析技术[11]。它所解决的问题是：当 n 个对象中各对象之间的相似性能够确定时，从这种相似性给出的信息出发，在较低维的空间中将这 n 个对象的几何体图形描绘出来，使其尽可能与原先的相似性大体匹配，并使得由降维所引起的任何变形达到最小，从而最大程度上揭示这 n 个客体之间的真实结构关系。由于多维空间 $\mathbb{R}^k$ 中排列的每个点代表一个对象，因此点间的相似性高度相关。也就是说，两个相似的对象由多维空间中两个距离相近的点表示，而两个不相似的对象由多维空间两个距离较远的点表示。

回归测试子集选择的来源是原测试集 T。若将 T 以它的执行剖面集表示，则它可看作是由 n 个多维对象组成的集合。软件发生变更，则 T 对应的执行剖面集也将随之改变。在无法完全重测 T 的情况下，利用多维标度法计算对测试更改有贡献的测试用例执行剖面数据，同时依靠 MDS 的安全降维特性，将这些高维对象在二维空间 $\mathbb{R}^2$ 中以几何点的形式展示出来。其中，距离相近的点就表示对变更的测试贡献程度相似的测试用例，距离较远的点则代表了对变更的测试程度差异较大的用例。从而，测试人员能够按照测试程度的不同从 $\mathbb{R}^2$ 中直观地选择典型用例。这样，最终得到的回归测试子集 T' 的测试结果能够等效 T，说明 P' 的质量。

3 测试用例选择

3.1 方法框架

如果软件发生了变更，那么变更或受变更影响[4,5]的软件部分最可能存在缺陷，这是由于未被执行的程序事件是不可信的。因此构成测试执行剖面的程序事件，通常由测试人员关心的变更或受变更影响的程序元素组成，如语句、方法的执行，数据流、控制流的流转等。当 T 执行 P 后，就可得到对应的执行剖面集合 PF。

不考虑软件的新增功能，变更测试需求 R 中的具体测试需求条目 r_i 通常由变更或受变更影响的程序元素组成。也就是说，R 与 PF 的组成成分都是测试人员关注的程序元素。那么，将每个执行剖面由各 r_i 所对应的程序元素表示，则各测试用例对 R 的测试情况可以矩阵的形式得以体现。基于覆盖的方法选出的回归测试子集，通常仅满足将 R 中各 r_i 对应的程序元素都覆盖到，并不考虑各个测试用例之间的关系，无法反映各用例的测试目的，从而难以等效 T 的测试效果。若充分利用执行剖面所蕴含的软件行为信息，通过计算 PF 中各用例执行剖面之间的距离，则能够得到那些测试用例具有的类似的测试目的和变更覆盖效果，并用多维标度法在二维空间展示出来，使测试人员可以显式地看到这一结果，从而便于测试人员参与回归测试集合的选择过程，增加选择的准确性。方法具体框架如图 1 所示。

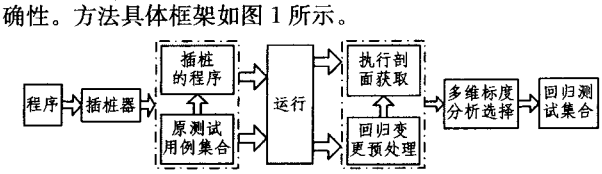

图 1 方法框架

最初测试时，可由原测试集合 T 的运行获得它的执行剖面集 PF。当软件变更后，需要分析软件结构的变化，将 PF 做相应的变更预处理，获得变更执行剖面集 PF_D，并将 PF_D 作为矩阵计算它的距离矩阵 T_D，接着计算 T_D 组成的距离阵 D_{TD}，得到对变更有贡献的测试用例在 $\mathbb{R}^2$ 中的分布，并使用失效追踪的选择方式辅助测试人员直观地选择所需的回归测试集合 T'。

3.2 变更预处理

将原测试用例集以向量 $T=(t_1,t_2,\cdots,t_n)$ 表示，它的执行剖面集为向量 $PF=(pf_{t_1},pf_{t_2},\cdots,pf_{t_j},\cdots,pf_{t_n})'$。$PF$ 由测试人员关注的程序事件向量 E 表示，$E=(\hat{e}_1,\hat{e}_2,\cdots,\hat{e}_p)$，其中 $\hat{e}_i$ 为程序元素随机变量。pf_{t_j} 由 E 中各随机变量的执行次数或覆盖情况决定，即 $pf_{t_j}=(e_{t_j1},e_{t_j2},\cdots,e_{t_ji},\cdots,e_{t_jp})$，其中 e_{t_ji} 为元素 $\hat{e}_i$ 的执行次数或覆盖情况。由此，可以得到 T 的执行剖面集 PF_T 为

$$PF_T=\begin{Bmatrix} pf_{t_1} \\ \vdots \\ pf_{t_j} \\ \vdots \\ pf_{t_n} \end{Bmatrix}=\begin{Bmatrix} e_{t_11} & \cdots & e_{t_1i} & \cdots & e_{t_1p} \\ \vdots & & \vdots & & \vdots \\ e_{t_j1} & \cdots & e_{t_ji} & \cdots & e_{t_jp} \\ \vdots & & \vdots & & \vdots \\ e_{t_n1} & \cdots & e_{t_ni} & \cdots & e_{t_np} \end{Bmatrix}$$

程序发生变更，则 E 就由变更或受变更影响的程序元素组成，即 $E'=(\hat{e}'_1,\hat{e}'_2,\cdots,\hat{e}'_q)$。以 PF_T 为蓝本做影响域分析，将各用例 t_i 的执行剖面转为 $pf'_{t_i}=(e'_{t_i1},e'_{t_i2},\cdots,e'_{t_ii},\cdots,e'_{t_iq})$，从而 T 的执行剖面集向量 PF_T' 为：

$$PF'_T=\begin{Bmatrix} pf'_{t_1} \\ \vdots \\ pf'_{t_j} \\ \vdots \\ pf'_{t_n} \end{Bmatrix}=\begin{Bmatrix} e'_{t_11} & \cdots & e'_{t_1i} & \cdots & e'_{t_1q} \\ \vdots & & \vdots & & \vdots \\ e'_{t_j1} & \cdots & e'_{t_ji} & \cdots & e'_{t_jq} \\ \vdots & & \vdots & & \vdots \\ e'_{t_n1} & \cdots & e'_{t_ni} & \cdots & e'_{t_nq} \end{Bmatrix}$$

比较 PF_T 和 PF'_T，可能存在测试用例 t_i，当 $\hat{e}'_i=\hat{e}_i$ 时，它的执行剖面 pf'_{t_i} 对 $\hat{e}'_i$ 的执行次数或覆盖情况都与 pf_{t_i} 对 $\hat{e}_i$ 的相应情况相同。这样的用例对变更测试需求没有贡献，不作为回归子集的候选用例，从 T 中排除。为了明确，将 E' 重定义为 $E_D=(\hat{e}'_{D1},\hat{e}'_{D2},\cdots,\hat{e}'_{Dq})$，得到对变更有贡献的测试用例集 T_D 的变更执行剖面集为

$$PF_D=\begin{Bmatrix} pf_{t_{D1}} \\ \cdots \\ pf_{t_{Dj}} \\ \cdots \\ pf_{t_{Dm}} \end{Bmatrix}=\begin{Bmatrix} e_{t_{D1}D1} & \cdots & e_{t_{D1}Di} & \cdots & e_{t_{D1}Dq} \\ \cdots & & \cdots & & \cdots \\ e_{t_{Dj}D1} & \cdots & e_{t_{Dj}Di} & \cdots & e_{t_{Dj}Dq} \\ \vdots & & \vdots & & \vdots \\ e_{t_{Dm}D1} & \cdots & e_{t_{Dm}Di} & \cdots & e_{t_{Dm}Dq} \end{Bmatrix}$$

式中，$m\leqslant n$。

基于覆盖的回归测试选择[2,4,5,8]方法通常按照覆盖 $\hat{e}'_{Di}$ 的多寡选择测试用例，既不考虑该元素的执行次数，也不考虑各用例之间的关系，忽视了用例的测试目的。本文提出的方法通过执行剖面展现测试用例的行为特征，从而反映测试目的，并通过计算各用例剖面间的距离来衡量用例的测试程度相似情况。

3.3 距离矩阵获取

全面分析 T_D 中各用例对程序变更的测试情况，需要表征各用例测试软件变更能力的亲疏程度，因此计算 t_{Di} 与 t_{Dj} 之间的距离 d_{ij}，构建距离矩阵 D_{TD}。由于 $e_{t_{Dj}Di}$ 可为该元素的执行次数，为避免大数据属性值影响整个距离公式，而导致其它属性被忽略，因此在计算距离时，需要使用极差对各属性值进行标准化处理。

定义 2(极差)　设矩阵 $M'=\begin{Bmatrix} x_{11} & \cdots & x_{1j} & \cdots & x_{1l} \\ \vdots & & \vdots & & \vdots \\ x_{i1} & \cdots & x_{ij} & \cdots & x_{il} \\ \vdots & & \vdots & & \vdots \\ x_{k1} & \cdots & x_{kj} & \cdots & x_{kl} \end{Bmatrix}$，则 $R_j=\max\limits_{i=1,\cdots,k}x_{ij}-\min\limits_{i=1,\cdots,k}x_{ij}$，$(j=1,\cdots,l)$，$R_j$ 称为极差。

d_{ij} 采用欧氏距离，标准化处理后 $dst_{Dij}=\sqrt{\sum\limits_{k=1}^{q}\left(\frac{e_{t_{Di}Dk}-e_{t_{Dj}Dk}}{R_{Dk}}\right)^2}$，$i,j=1,2,\cdots,m$。从而可以得到 T_D 的剖面距离矩阵，如表 1 所列。其中 $\mathrm{dst}_{Dij}=\mathrm{dst}_{Dji}$，$i,j=1,2,\cdots,m$。有了距离矩阵，就可以进一步可视化测试集的测试效果。

表 1　T_D 剖面距离矩阵

	$D(t_{D1})$	$D(t_{D2})$	…	$D(t_{Dm})$
$D(t_{D1})$	0	dst_{D12}	…	dst_{D1m}
$D(t_{D2})$	dst_{D21}	0	…	dst_{D2m}
…	…	…	…	…
$D(t_{Dm})$	dst_{Dm1}	dst_{Dm2}	…	0

3.4 测试用例分类

定义 3(距离阵)　一个 $n\times n$ 阶矩阵 $D=(d_{ij})$，如果满足条件：(a) $D'=D$，(b) $d_{ij}\geqslant 0$，$d_{ii}=0$，$i,j=1,\cdots,n$，则称 D 为距离阵，d_{ij} 为第 i 个点与第 j 个点之间的距离。

可以看出，T_D 的距离矩阵就是它的 $m\times m$ 维距离阵 D_{TD}。有了距离阵，就可以在空间 $\mathbb{R}^k$ 上求距离阵中代表 m 个测试用例的点 $x_1,\cdots,x_m$，使得这 m 个点的欧氏距离与距离阵中的相应值在用例对程序变更测试程度相似的意义下尽量接近。为了使求得的结果易于测试人员理解，本文取空间维度 $k=2$。将第 i 个测试用例对应的向量记为 x_i，x_i 在 $\mathbb{R}^2$ 的坐标记作 (x_{i1},x_{i2})。为了在 $\mathbb{R}^2$ 上可视化测试用例，需要得到这 m 个向量的拟合构图。

定义 4(拟合构图)　将按要求求得的、表示 m 个测试用例的点 $x_1,\cdots,x_m$ 记为 $X=(x_1,\cdots,x_m)'$，则 X 为 D_{TD} 的一个解，称 X 为距离阵 D_{TD} 的一个拟合构图。

拟合构图的意义是得到这 m 个向量的坐标，在 $\mathbb{R}^2$ 作图，使得它们的距离阵和原始的 m 个测试用例的距离阵 D_{TD} 接近，用于解释这 m 个测试用例对软件变更的测试程度的相似性，以便测试人员选择适合的回归用例集 T'。

得到 D_{TD} 拟合构图的步骤如下：

(1)根据距离矩阵 D_{TD}，得到内积矩阵元素 b_{ij}，其中，

$$b_{ij}=\frac{1}{2}\left(-d^2_{z_{Dij}}+\frac{1}{m}\sum_{i=1}^{m}d^2_{z_{Dij}}+\frac{1}{m}\sum_{j=1}^{m}d^2_{z_{Dij}}-\frac{1}{m^2}\sum_{i=1}^{m}\sum_{j=1}^{m}d^2_{z_{Dij}}\right)$$

(2) 作出矩阵

$$B=(b_{ij})_{m\times m}=\begin{bmatrix} (x_1-\bar{x})' \\ \vdots \\ (x_m-\bar{x})' \end{bmatrix}(x_1-\bar{x},\cdots,x_m-\bar{x})$$

式中，$\bar{x}=\frac{1}{m}\sum\limits_{i=1}^{m}x_i$。

(3)计算内积矩阵 B 的特征值 $\lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant \lambda_m$ 和 k 个最大特征值 $\lambda_1 \geqslant \lambda_2 \cdots \geqslant \lambda_k > 0$ 对应的单位特征向量，本文中定义 $k=2$。

(4)根据 $x=\Gamma\Lambda^{1/2}$ 计算 x，得到用例集 T_D 在二维空间 $\mathbb{R}^2$ 中的拟合构图。

3.5 回归用例选择

T_D 的拟合构图在图形上会将距离较近的测试用例以点团的形式表现出来，这就意味着这些测试用例对软件更改部分的测试程度是类似的。那么在挑选回归测试集合时，就可以从各个不同的点团中挑选，以等效 T 的测试效果。另外，研究[7]发现失效测试用例的分布通常具有这些特征：1)孤立；2)分布距离通常较近；3)有可能越过分类界限呈链状分布。因而，可以采用失效追踪的抽取方法获得 T'。具体步骤如下：

(1)从每个点团中随机抽样一个测试用例，审核其是否失效。

(2)如果某个点对应的测试用例发现了缺陷，选择与该点距离最近的 w 个测试用例审核。

(3)如果步骤(2)选出的测试用例同样发生了失效，重复步骤(2)，直到找不出失效的测试用例为止。

(4)将得到的测试用例组成回归测试子集 T'。

从以上步骤看出，测试用例跟踪迭代不会局限于点团的限制，能够满足失效测试用例的分布特性。其中 w 可以根据需要任意选择。

4 实例检验

4.1 目标软件

本文采用 grep 作为目标软件。它隶属于 Gnu 工具集，使用正则表达式搜索文本，并把匹配的行打印出来，是功能完善的文本搜索工具。grep 共有 10068 行，由 146 个函数组成。为了验证多维标度法的正确性，我们人为在其中植入了 15 个缺陷，作为回归测试版本，并使用基于覆盖的选择和基于多维标度分析的方法分别获得各自的回归测试集，比较两者发现缺陷的数目，以验证多维标度法的有效性。由于采用多维标度法和覆盖法选择回归测试子集需要大量计算，单纯依赖人力难以在短时间内完成。基于此，研制了用于数据分析及聚类抽样的测试用例选择工具原型 Techoose。该工具提供的主要功能为：1)将测试用例执行剖面集构成的数据矩阵作为输入，把矩阵数据处理为标准化矩阵数据；2)将标准化数据矩阵作为输入，按照需要做多元统计分析，得到分类后数据；3)以分类后数据作为输入，从每组中选择所需用例，得到选出的测试子集；4)将测试用例执行剖面数据矩阵作为输入，按照覆盖多少顺序选择测试用例，得到选出的测试子集。

4.2 实验检验及分析

植入的 15 个缺陷分别位于 15 个不同的函数中，共 808 个用例对其有测试贡献，构成 808 行 15 列的变更执行剖面集 PF_D，标准化处理后得到距离矩阵 D_{TD}，如表 2 所列。

表 2 距离矩阵 D_{TD}

	case1	case2	…	case807	case808
case1	0.000	0.859	…	0.272	0.222
case2	0.859	0.000	…	1.074	1.012
…	…	…	…	…	…
case807	0.272	1.074	…	0.000	0.062
case808	0.222	1.012	…	0.062	0.000

计算 D_{TD} 的拟合构图，各测试用例对应的点在二维空间 $\mathbb{R}^2$ 中的坐标如表 3 所列。

表 3 各点坐标

	Dimension	
	1	2
Case 1	.283	−.321
Case 2	−.099	.449
Case 3	.195	−.104
Case 4	.195	−.104
Case 5	.062	.900
…	…	…
Case 807	.205	−.581
Case 808	.188	−.521

它们在二维空间中的分布如图 2 所示，图中在外围散布的点是执行剖面较为独特的测试用例对应的几何点，也就是说，这些测试用例引发了其它测试用例难以引发的程序事件，应选入回归测试子集。另外，聚集较密的测试用例点团，代表它们的执行剖面较为相似，意味着这些测试用例可能在功能方面的测试目的较为近似，从而使得对软件执行引发的事件比较类似。对它们采用上述回归用例选择方式，从每个点团中随机抽样。如果发现缺陷，则寻找该点距离最近的一个点，审核是否发现缺陷。如果发现，则继续寻找离该点最近的点，直到找不到缺陷为止。基于覆盖的回归子集的选择使用贪心算法，覆盖变更最多的测试用例最先选择，接着选择覆盖余下未覆盖变更最多的测试用例。如果覆盖的变更数目相同，选择首先读到的测试用例。

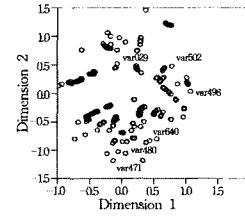

图 2 多维标度在 $\mathbb{R}^2$ 的分布图

我们在分析那些散布的点时，就发发现了 6 个缺陷，如图 2 所示。其中 var 指代测试用例。对密集的测试用例点团进行多次抽样计算后，平均 300 个测试用例左右即可发现 15 个缺陷。而基于覆盖的测试用例选择策略则要到 600 多个测试用例后才能发现所有缺陷。多维标度分析与覆盖选择得到的回归测试子集对比如图 3 所示。

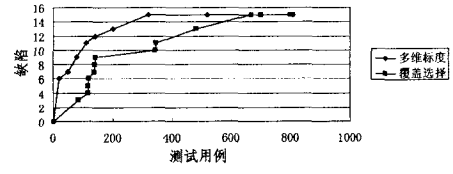

图 3 多维标度与覆盖方法缺陷发现对照

比较两种方法获得的回归测试子集发现缺陷的情况可以看出，多维标度法选择出的测试子集效率明显高于基于覆盖的方法选出的子集。另外，多维标度法能够直观地看到测试执行剖面的分布情况，因而选择出的测试子集不仅能够代表原测试集的总体目的，而且能够方便地挑出测试效果较为独特、引发了不常见程序事件的测试用例。

结束语 回归测试用例子集的选择是测试人员时常需要

(下转第 140 页)

框架之内，构造了基于文化的最大-最小蚁群算法，并用来解决最优路径选择问题。此计算模型在知识和群体层面使用双重进化机制支持问题的求解和知识的提取，充分利用了种群的进化机制和知识的指导作用，在很大程度上提高了种群的多样性及收敛速度，达到了防止早熟、降低计算代价的目的。通过大量的实验验证及与其他算法的比较得出，C-MMAS算法求解速度快、寻优成功率高，是一种求解复杂组合服务选择问题的有效方法。

参 考 文 献

[1] Curbera F, et al. Unraveling the web services: An introduction to SOAP, WSDL, and UDDI[J]. IEEE Internet Computing, 2002, 6(2): 86-93

[2] Web Service — Business Process Execution Language (WS BPEL), Version 2.0[S]. OASIS Committee Draft, 17th May, 2006

[3] Patil S, Newcomer E. ebXML and Web services[J]. IEEE Internet Computing, 2003, 7(3): 74-82

[4] Garey M R, Johnson D S. Computers and Intractability: A Guide to the Theory of NP-Completeness[M]. New York: Freeman Press, 1979

[5] Zeng L, Benatallah B, Ngu A H H, et al. QoS-aware Middleware for web services composition[J]. IEEE Transaction on Software Engineering, 2004, 30(5): 311-327

[6] Ardagna D, Pernici B. Global and local QoS guarantee in Web service selection[C]// Proc. of Business Process Management Workshops. 2005: 32-46

[7] 刘书雷，刘云翔，张帆，等. 一种服务聚合中QoS全局最优服务动态选择算法[J]. 软件学报，2007，18：646-656

[8] Canfora G, Penta M D, Esposito R, et al. An approach for QoS-aware service composition based on genetic algorithms[C]// Proceedings of Genetic and Evolutionary Computation Conference (GECCO 2005). 2005: 1069-1075

[9] Yu T, Lin K J. Service selection algorithms for Web services with end-to-end QoS constraints[C]// Proceedings of IEEE International Conference on E-Commerce Technology (CEC). 2004: 129-136

[10] 彭晓明，何炎祥，朱兵舰. 蚁群算法在Web服务组合中的应用[J]. 计算机工程，2009，35：182-187

[11] 张成文，苏森，陈俊亮. 基于遗传算法的QoS感知的Web服务选择[J]. 计算机学报，2006，29：1029-1037

[12] Thomas S, Holger H, et al. Max-Min ant system [J]. Future Generation Computers System, 2000, 16(8): 889-914

[13] Reynolds R G. An Introduction to Cultural Algorithms [C] // Proceedings of the Third Annual Conference on Evolutionary Programming. River Edge, New Jersey: World Scientific, 1994: 131-39

[14] 刘升，王行愚，游晓明. 求解TSP问题的文化蚁群优化算法[J]. 华东理工大学学报，2009，35(2)：288-292

[15] 郭一楠，王辉. 文化算法研究综述[J]. 计算机工程与应用，2009，45(9)：41-46

[16] Cardoso J, Sheth A P, Miller J A, et al. Modeling quality of service for workflows and Web service processes[J]. Web Semantics Journal: Science, Services and Agents on the World Wide Web Journal, 2004, 1(3): 281-308

[17] Jaeger M C, Rojec-Goldmann G, Muhl G. QoS aggregation for Web service composition using workflow patterns[C]// Proceedings of IEEE International Conference on Enterprise Distributed Object Computing (EDOC). 2004: 149-159

[18] Dorigo M. Ant colony system: a cooperative learning approach to the traveling salesman problem [J]. IEEE Transactions on Evolutionary Computation, 1997, 1(1): 53-56

(上接第134页)

面临的难题。以往常用的基于变更覆盖的选择方法，由于仅考虑了测试用例覆盖变更的多少，没有考虑测试用例对变更测试的典型性，使得最终选择出的回归测试子集难以具有原测试集等效的测试效果。利用多维标度法获得的回归测试集合则解决了这一问题，使得最终获得的回归测试子集不仅检验了软件变更，而且代表了原测试集不同的测试目的，从而使测试结果能够代表原测试集的检验效果，能够说明更改后软件的质量。

参 考 文 献

[1] Kim Jung-Min, Adam P, Gregg R. An empirical study of regression test application frequency[J]. Software Testing, Verification & Reliability, 2005, 15(4): 257-279

[2] Leon D, Podgurski A. A Comparison of Coverage-based and Distribution-based Techniques for Filtering and Prioritizing Test Cases[C]// Proceedings of the 14th International Symposium on Software Reliability Engineering. 2003: 442-454

[3] Leon D, Masri W, Podgurski A. An Empirical Evaluation of Test Case Filtering Techniques Based on Exercising Complex Information Flows[C]// Proceedings of the 27th International Conference on Software Engineering. 2005: 412-421

[4] Dennis J, Neelam G. Experiments with Test Case Prioritization Using Relevant Slices[J]. The Journal of Systems and Software, 2008, 81(2): 196-221

[5] Dennis J, Neelam G. Test Case Prioritization Using Relevant Slices[C]// Proceedings of the 30th Annual International Computer Software and Applications Conference. 2006: 411-420

[6] Dickinson W, Leon D, Podgurski A. Finding Failures by Cluster Analysis of Execution Profiles[C]// Proceedings of the 23rd International Conference on Software Engineering. 2001: 339-348

[7] Cesin L, Zaen D. Profile Analysis Techniques for Observation-based Software Testing. 2005: 132

[8] Amitabh S, Jay T. Effectively prioritizing tests in development environment[J]. ACM SIGSOFT Software Engineering, 2002, 27(4): 97-106

[9] 林亚平，胡玉鹏，陈治平. 基于执行剖面过滤的分割测试[J]. 湖南大学学报，2006，33(1)：110-113

[10] 赵亮，王建民，孙家广. 软件测试准则的有效性度量研究[J]. 计算机研究与发展，2006，43(8)：1457-1463

[11] 朱建平. 应用多元统计分析(第一版)[M]. 北京：科学出版社，2006：213