文章编号: 1000-6893(2006)05-0882-05

基于自组织神经网络的软件功能测试数据自动生成

傅 博
(北京航空航天大学 工程系统工程系, 北京 100083)

Automatic Software Functional Test Data Generation Based on Dynamic Self-organizing Neural Networks

FU Bo
(Department of System Engineering of Engineering Technology, Beijing University of Aeronautics and Astronautics, Beijing 100083, China)

摘 要: 针对面向软件功能的测试数据自动生成问题,提出了一种动态自组织特征映射方法,用于生成揭示软件功能故障的测试数据(简称故障数据)。该方法主要有两部分组成, ¹ 采用具有全局多峰搜索特性的小生境遗传算法,在输入空间内搜索功能测试数据,生成少量的初始故障数据; º 由初始故障数据,采用具有联想和分类能力的可变结构自组织特征映射,不断迭代生成大量相近而不同的故障数据,以便给开发者提供引发这些软件故障的信息,从而确定软件故障行为的模式或假设。用某型空空导弹发射控制软件进行了实验,运行结果表明了方法的有效性,故障数据生成效率高于遗传算法和随机法。

关键词: 软件测试; 自组织特征映射; 神经网络; 小生境遗传算法; 测试数据自动生成
中图分类号: TP311.5 文献标识码: A

Abstract: The automatic test data generation for program functions is one of the elementary problems in software functional testing. This paper addresses the problem by presenting a technique of dynamic self-organizing neural networks to automatically generate test data for revealing software faults. The technique consists of two parts: the first one is niche genetic algorithm, which generate a small initial fault test data set in software input domain; the second one is dynamic self-organizing feature map, which can repeatedly generate lots of test data for finding faults by using initial fault test data set. These can provide the developer with fault data information to identify fault patterns or hypothesis in software. The approach is used on a C program which is part of missile launch control system. Experimental results show that the method is more efficient than niche genetic algorithms and random techniques.

Key words: software test; self-organizing feature map; neural network; niche genetic algorithm; automatic test data generation

软件测试数据自动生成是软件测试领域中的关键技术难题之一,可分为结构和功能两类。功能测试数据自动生成是依据需求规格说明,抽取测试功能,按照一定的方法生成测试数据。一个好的功能测试数据应是一个对以前未被发现的缺陷有高发现率的数据,而不是一个表明程序功能正确的数据。所以,功能测试数据自动生成是对发现故障的测试数据生成的研究。文献[1]指出,当失效发生时,有必要采用相似而不同的发现失效的测试数据进行集中测试,以支持故障隔离,从而实现故障纠正。文献[2]首先选择一些测试数据的初始样本,评价这些样本并记录失效样本,然后生成和失效样本相近而不同的测试数据。文献[3]在故障场景群体中,生成相近的故障数据。受文献[2,3]的启发,文献[4]采用遗传算法(GA)并引入小生境生成相似而不同的故障数据。适应值函数包含两个因素: 一是运行发生概率,表达了用户使用系统的场景; 二是失效强度,揭示了系统中的失效情况。这种方法适合于已知发生概率和失效强度,并且存在失效种类较多的情况。但在实际应用中,事先确定这些值比较困难,软件存在的失效种类较少, GA 搜索这些失效变成低效率的随机搜索。文献[5]采用 GA 进行功能测试数据的自动生成,适应值函数包括 3 部分: 新颖度、相似度和严重度。新颖度指测试数据与过去测试数据的相异性。相似度指测试数据与故障测试数据的相似程度。严重度指测试数据导致故障的严重程度。这种方法适合于错误种类较多且已知严重度的情况。文献[6]采用模拟退火算法选择测试数据,以发现软件规格的错误和程序的条件异常。

收稿日期: 2005-05-09; 修订日期: 2005-08-15

将前置条件2非后置条件写成析取范式, 算法通过搜索析取范式为零, 得到故障测试数据。该方法虽然能够较快搜索到满足前置条件项, 但由于程序错误的不可预知性, 对程序错误导致的非后置条件项的搜索, 则退化为随机搜索。文献[7] 由专用工具生成测试数据集, 采用 BP 神经网络作为分类器, 通过对该测试数据集的学习, 能够预测专用工具新生成测试数据的揭错能力。BP 神经网络不能生成测试数据, 而且测试数据训练过程比较复杂, 神经网络自动生成测试数据的研究还没有看到。

本文对神经网络自动生成功能测试数据进行研究, 提出一种小生境遗传算法(NGA) 和自组织特征映射(SOFM) 相结合的动态生成方法, 简称动态自组织特征映射(DSOFM) 。DSOFM 能够生成相近而不同的发现错误的测试数据, 这些故障数据提供引发这些软件故障的信息, 容易确定系统故障模式。以某型空空导弹发射控制软件进行了实验, 运行结果验证了方法的可行性和有效性, 生成故障数据的效率高于小生境遗传算法和随机法。

1　基于 DSOFM 的功能测试数据生成

(1) 自组织特征映射(SOFM)[8]

SOFM 是一种具有侧向联想能力的神经网络, 可以在无监督的情况下, 从输入数据中找出有意义的规律。SOFM 能把多维输入空间映射成输出层神经元的一维或二维阵列, 构成一个存在有意义的拓扑序列的输出空间。SOFM 的结构如图 1 所示, 输入层结点数对应于输入空间的维数; 输出层节点数对应于映射后的模式空间维数, 按照一定的拓扑连接关系和邻域函数, 经自组织算法训练后, 不同结点代表不同的分类模式。这些特征使 SOFM 适用于软件功能测试数据和故障数据的分类和生成。

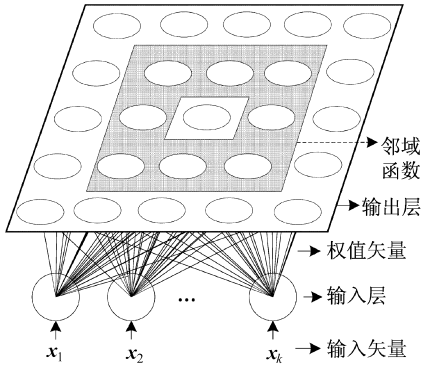


图 1　SOFM 神经网络结构

Figl 1　Architecture of SOFM neural network

(2) 软件功能测试数据生成概念　为便于说明 DSOFM 进行软件功能测试数据生成, 对软件功能测试作如下定义。

定义 1　设被测程序功能空间中的某一功能, 存在输入2输出对(A, B) , 满足 if(A) then(B) 条件, 则程序功能记作 F(A, B) 。对于功能 F(A, B) , 定义谓词 C(A, B) = A C ¬ B, 则有 if C(A, B) then ¬ F(A, B) , 则称¬ F(A, B) 为程序功能故障。

定义 2　对于 F(A, B) , 存在一个输入参数 x_i (x_1, x_2, , , x_k) 和相应输出参数 y_i (y_1, y_2, , , y_q) , 使得 C(A, B) 为真, 则称 x_i 是一个发现程序功能故障的测试数据, 简称故障数据, n 个故障数据 x_i 组成故障数据集 x 。i= 1, 2, , , n, x_i I x。

程序功能及其故障数据集表达如图 2 所示。DSOFM 的目标就是在程序的输入空间内, 生成测试数据并实施功能测试, 集中搜索故障数据集。

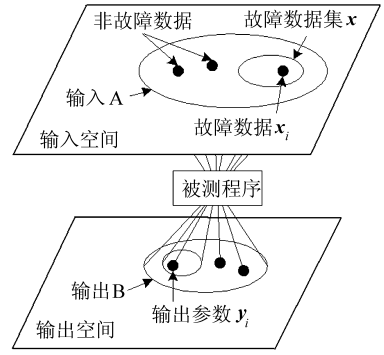


图 2　程序功能 F(A, B) 及其故障数据

Figl 2　Function F(A, B) and its fault data set

(3) 基于 DSOFM 的测试数据自动生成

1　基于 DSOFM 的测试数据自动生成原理

生成原理模拟了人脑搜索和辨别感兴趣的信息的过程。首先在整个可行域内作宏观搜索, 一旦发现所需信息, 则抽取信息特征并进行联想, 从而在局部进行详尽的微观搜索, 直到得到满意的结果为止。生成原理如图 3 所示, 分为两部分: 一是在输入空间内, 采用 NGA 搜索功能 F(A, B) 的输入 A, 用 A 输入被测程序, 发现程序故障, 生成初始故障数据集 x(0) ; 二是由 x(0) , 经过 SOFM 生成相近的测试数据集 xc(0) , 运行被测程序, 搜索程序故障, 生成故障数据集 xd(0) ; 然后令 x(1) = xd(0) , 进入下一个循环, 当满足条件 T 时, 则流程结束并输出故障数据集 xd, 条件 T 是指特征映射没有变化或到达最大训练次数。因| xd| \ | x| , 则 SOFM 结构需要不断增加, 这种动态 SOFM 结构能够生成不断增大的故障数据集。第 1 部分

采用 NGA 随机搜索 x(0) , 模拟了人脑的宏观搜索; 第 2 部分采用 SOFM 生成满意的故障数据集, 模拟了人脑的微观搜索。所以, DSOFM 自动生成测试数据过程属于智能化的搜索过程。

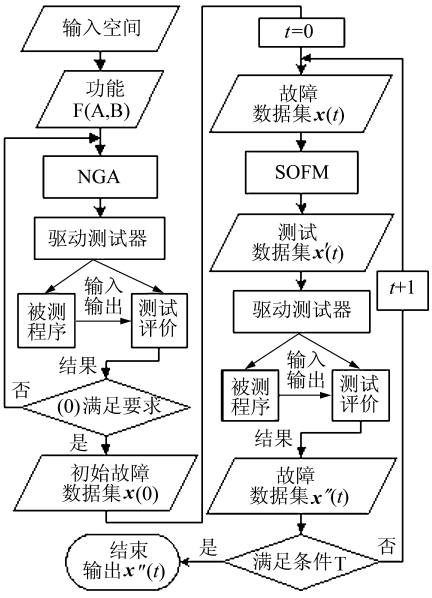


图 3 基于 DSOFM 测试数据自动生成

Figl 3 Test data generation based on DSOFM

º 基于 NGA 的初始故障数据生成

NGA 是在基本 GA 的基础上, 不断调整种群中个体的适应度, 使种群在进化过程中能够依据调整后的适应度进行选择, 以维护种群的多样性, 构造出小生境的进化环境。NGA 是一种多峰搜索问题的优化算法, 使得能够搜索到多个全局最优解。根据功能 F(A, B) , 通过设计适应值函数, 使得在输入空间内搜索输入 A, 即生成测试数据, 并在 A 内发现初始故障数据。基于 NGA 的初始故障数据生成算法描述如下:

l 设置进化代数计数器 t1 z 1; 在输入空间内随机生成 m 个个体 xj(x1, x2, , , xk) 组成初始群体 P(t) , j= 1, 2, , , m, 并计算个体的适应度: Qj(A, B) = [f1(A) + f2(Cj(A, B)) + E] - 1。函数 f2 由测试评价模块实现, 当 Cj(A, B) 为真时, f2= 0, 为假时, f2= H, f1 为接近满足输入 A 的条件的程度, 当满足条件 A 时, f1= 0; E和 H为非零常系数。

m 依据各个个体的适应度 Qj(A, B) 进行降序排序, 记录前 n 个个体 x1, x2, , , xn 组成 x(0) , n= | x(0) | , n< m。

n 遗传操作。对群体 P(t1) 进行比例选择运算得到 Pc(t1) ; 对 Pc(t1) 进行交叉运算, 得到 Pd(t1) ; 在输入空间内对 Pd(t1) 进行均匀变异运算, 得到 PÊ(t1) 。

o 小生境操作。将第 n 步得到的 PÊ(t1) 的 m 个个体和第 m步记录的 n 个个体合并, 得到含有 m+ n 个个体的新群体。计算新群体中每两个个体 xu(x1, x2, , , xk) 和 xv(x1, x2, , , xk) 之间的欧氏距离为

$$+ x_u - x_v + = \sqrt{\sum_{l=1}^{k} (x_{u,l} - x_{v,l})^2}$$

$$u = 1, 2, , , m + n - 1$$

$$v = u + 1, u + 2, , , m + n$$

设给定小生境距离为 L, 当 + xu - xv + < L 时, 则比较 xu 和 xv 适应度大小, 对适应度较低的个体处以罚函数 Qmin(xu, xv) 。

p 根据此 m+ n 个个体的新适应度大小对其进行降序排序, 记录前 n 个个体 x1, x2, , , xn。

q 终止条件判断。设终止条件 R= (Gmax D Sn) , 其中 Gmax 表示最大进化代数, Sn 表示前 n 个个体均满足 C(A, B) 条件, 即 x(0) 满足要求。若 R 为假, 则更新进化代数计数器 t1 z t1+ 1, 并将第 p 步排序中的前 m 个个体作为新的下一代群体 P(t1) , 然后转第 n 步; 若 R 为真, 则输出 x(0) , 算法结束。

NGA 通过在适应值函数中引入 f1(A) , 在输入空间内实施均匀变异, 交叉操作, 从而生成功能测试数据。通过在适应值中引入 f2(C(A, B)) 和小生境操作, 实现程序功能测试并生成初始故障数据。

» SOFM 生成测试数据和故障数据

SOFM 通过自适应网络结构, 多次联想和分类循环, 生成测试数据和故障数据。设某一循环 t 的故障数据集 x(x1, x2, , , xn) , 存在 x 中的 xj (x1 x2 , xk) , j= 1, 2, , , n, 则 SOFM 输入矢量为 xj = [x1 x2 , xk] T, 输入层神经元数为 k。设输出层神经元为如图 1 所示的 N @N 二维方阵点列, 则输出层神经元总数为 N2。xj 与输出层神经元 i 相应的权值矢量 wi = [wi1 wi2 , wik] T, i= 1, 2, , , N2, 且 N2> n。由于每个循环 SOFM 的 N 随着| x| 的不同而发生变化, 所以必须确定 k 和 N 的关系, 以便实现 SOFM 的动态自适应结构。在 t 循环 SOFM 的数据生成算法描述如下:

l 动态确定 SOFM 结构。输入层神经元数为 k, 则输出层神经元总数为 N2。其中放大函数 N= [Ak0l5+ B] , A和 B为大于 1 的放大系数, 并设

初值为 A_0 和 B_0。

m 初始化权值矢量 $w_i(0)$ 为较小随机值, 初始化学习率 $G(0)$ 和邻域函数 $r(0)$ 均为较大值。

n 将故障数据集 x 作为训练样本执行步骤 o, p 和 q 实施网络训练。

o 将数据集 x 中的 $x_j(x_1, x_2, \cdots, x_k)$ 输入到网络的输入层神经元, $j = 1, 2, \cdots, n$。

p 相似匹配, 选出权值矢量最匹配 x_j 的输出层神经元作为获胜神经元, 获胜神经元为 $i(x_j) = q$, 当 $\| w_q - x_j \| < \| w_i - x_j \|$, $i = 1, 2, \cdots, N^2$。

q 利用邻域函数 $r(t_2)$, 训练邻域内的权值矢量, 当 $i < r(t_0)$ 时, $w_i(t_2+1) = w_i(t_2) + G(t_2) \# [x_j - w_i(t_2)]$, t_2 为训练次数。

r 线性减小学习率 $G(t_2)$; 减小邻域函数 $r(t_2)$。

s 条件 T 判断。当特征映射没有明显变化或到达最大训练次数 $t_{2max}(0)$ 时终止, 记录权值集 $w(w_1, w_2, \cdots, w_{N^2})$, 否则转第 n 步。

t w 与 x 的余集 $x_c = w - x$ 即为新生成的测试数据集, 输入到被测程序, 得到新故障数据集 x_d。

u 更新循环计数和故障数据集, $t \leftarrow t+1$, $x \leftarrow x_d$, 进入下一个循环。

v 如果连续 Δt 次循环 $|x|$ 不发生变化, 则改变结构放大系数 $A(\Delta t) = A_0 + K\Delta t$; 或改变最大训练次数 $t_{2max}(\Delta t) = t_{2max}(0) + [D\Delta t]$, K 和 D 均为小于 1 的常数。

从上述可看出, 由于每个输出层神经元相连的权值矢量可以看做是神经元对输入矢量的响应的结果, 所以通过训练可以选出权值矢量与输入矢量有最小欧氏距离的输出层神经元作为获胜神经元。SOFM 不仅能使获胜神经元得到最大刺激, 而且该获胜神经元附近的一些神经元也将获得较大的刺激。通过为获胜神经元定义一个不断缩小的动态邻域函数, 使之与神经元排列的拓扑关系对应, 训练结果就能使邻近的神经元代表相似的类别, 从而实现侧向的联想能力, 生成相近的测试数据, 经运行被测程序生成故障数据。

SOFM 自适应网络结构通过放大函数实现。放大函数既要保证 SOFM 具有足够的侧向联想能力, 也要实现精确的模式分类, 又要考虑 SOFM 的训练效率, 所以需要设定合适的放大系数。在 SOFM 循环中, 如果故障数据集不发生变化, 可以通过调整放大系数来增加 SOFM 侧向联想能力, 或通过增加训练次数来提高模式分类的精度。

2　实　例

(1) 被测程序

以某型空空导弹发射控制软件为例, 该软件为嵌入式随动控制软件, 用于实时控制导弹瞄准、截获和发射。被测程序输入包括雷达解锁、发射指令、雷达投影、位标器、离轴角等参数, 输出包括目标截获、位标器阀门和 M1 DIS 等参数。在被测程序中引入两个功能错误: ¹ 在自主跟踪控制中引入小误差角错误; º 在雷达随动功能中引入随动算法错误。自主跟踪和雷达随动功能的输入数据经过离散化变换为整数, 输入空间为: $D = \{(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9) \mid x_1 \in [0, 1], x_2 \in [0, 1], x_3 \in [0, 1], x_4 \in [0, 1], x_5 \in [-19\,660, 19\,661], x_6 \in [-19\,660, 19\,661], x_7 \in [-19\,660, 19\,661], x_8 \in [0, 39\,320], x_9 \in [-19\,660, 19\,661]\}$。小误差角错误在输入空间某一区域内可导致目标截获故障, 随动算法错误在输入空间某一区域内可导致位标器启动阀门故障。

(2) 实验结果

被测程序为 C 语言程序, DSOFM 采用 C++ 语言实现, 分别对自主跟踪和雷达随动功能进行了 10 次实验。在 NGA 生成 $x(0)$ 中, 设 $n = |x(0)| = 10$, 取群体规模 $m = 50$, 单点杂交率为 0.8, 均匀变异率为 0.15。在基于 SOFM 的测试和故障数据生成中, 结构放大系数 $A_0 = 1.5$, $B_0 = 2$, $K = 0.02$, 循环数 $t = 200$。在一个循环中, 初始化学习率 $G(0) = 0.9$, 经过 $t_{2max}(0) = 2000$ 次训练后, 学习率 $G(2000) = 0.005$。

对于自主跟踪功能, 平均生成 73 个故障数据, 耗时 11.851 s, 被测程序运行次数为 5 543 次。生成单个故障数据的时间效率为 0.025 4 s, 程序运行效率为 76 次。对生成的 73 个故障数据进行分析, 不难找到故障数据规律, 得出小误差角判断错误的故障信息。

对于雷达随动功能, 平均生成 80 个故障数据, 耗时 3 s, 被测程序运行次数为 33 943 次。生成单个故障数据的时间效率为 0.038 s, 程序运行效率为 424 次。对生成的 80 个故障数据进行分析, 不难找到交叉耦合计算错误的故障信息。

另外, 本文分别对基于 NGA 的故障数据生成和均匀随机法的故障数据生成进行了实验, 实验结果及其和 DSOFM 结果的比较见表 1 和表 2。

表 1 自主跟踪功能的每个故障数据的生成效率

Table 1 Track function fault data generation efficiency

方 法	运行/次	时间/s
随机法	32 560	01 076 4
NGA	547	01 039 5
DSOFM	76	01 025 4

表 2 雷达随动功能的每个故障数据的生成效率

Table 2 Slave function fault data generation efficiency

方 法	运行/次	时间/s
随机法	624 657	11 119
NGA	3 387	01 154
DSOFM	424	01 038

(3) 结果分析

从表 1 和表 2 不难看出, NGA 的故障数据的生成效率(时间效率和运行效率) 高于随机法, 而 DSOFM 高于 NGA。随机法具有操作简单、容易实现等特点, 但由于缺乏对功能输入和故障数据模式的启发搜索能力, 所以必须以大量运行被测程序为代价, 当随着输入空间的增大和(或) 被测程序运行时间增长, 则随机法的搜索效率降低, 甚至变得不可行。NGA 具有对功能输入的启发搜索能力, 并通过维持小生境可以得到相近而不同的测试数据。但是 NGA 需要维持具有一定数量的群体规模进行演化, 当小生境规模变大时, 群体规模也必须相应变大。由于对每个个体适应值的计算均需要运行被测程序, 所以当群体规模变大时, NGA 运行效率降低。DSOFM 采用 NGA 生成较少的初始故障数据, 从而避免了 NGA 的缺点, 有效利用了 NGA 在输入空间内的启发式多峰搜索能力的优点。由于 DSOFM 的训练阶段不需要运行被测程序, 学习结束后生成的测试数据具有故障数据的部分特征, 包含故障数据的可能性较大, 所以用这些测试数据运行被测程序, 将以较高运行效率生成故障数据。

3 结 论

提出了一种应用于软件功能测试中的基于自组织神经网络的测试数据自动生成方法。该方法以发现程序功能故障为导向, 采用小生境遗传算法搜索少量导致功能故障的测试数据集。在此基础上, 利用自适应结构的自组织特征映射神经网络的分类和联想能力, 自动生成大量新的揭示功能故障的测试数据集。该方法有效模拟了人类搜索发现问题的能力, 具有无监督学习、搜索效率高等优点。导弹发射控制实例说明了该方法实现软件功能测试数据自动生成的可行性和有效性, 生成效率高于小生境遗传算法和随机法。下一步的工作是开发基于遗传算法和神经网络的测试数据生成工具, 用于实际软件系统的功能测试。

参 考 文 献

[1] Myers G J. The art of software testing[M]. New York: John Wiley and Sons, Inc, 1979.

[2] Dickinson W, Leon D, Podgurski A. Pursuing failure: the distribution of program failures in a profile space[C]// Proceedings of the 9th ACM SIGSOFT Symposium on Foundations of Software Engineering. 2001: 246-255.

[3] Schultz A C, Grefenstette J J, de Jong K A. Adaptive testing of controllers for autonomous vehicles[C]// Proceedings of the 1992 Symposium on Autonomous Underwater Vehicle Technology. 1992: 158-164.

[4] Patton R M. Application of intelligent methods for improved testing and evaluation of military simulation software[D]. Florida: University of Central Florida, 2002.

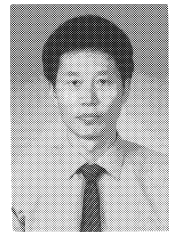
[5] Berndt D, Fisher J, Johnson L, et al. Breeding software test cases with genetic algorithms[C]// Proceedings of the Hawaii International Conference on System Sciences. Hawaii: [s. n.], 2003.

[6] Tracey N, Clark J, Mander K. Automated program flaw finding using simulated annealing[C]// Proc. Intcl Symp. Software Testing and Analysis, Software Engineering. 1998: 73-81.

[7] Andersson C, von Mayrhauser A, Chen T. Assessing neural networks as guides for testing activities[C]// IEEE Proceedings of METRICS'96. 1996: 155-165.

[8] Kohonen T. The self-Oranizing map[C]// Proceedings of the IEEE. 1990, 78(9): 1464-1480.

作者简介:



傅 博(1964-) 男, 河南睢县人, 高级工程师, 博士生。研究方向: 软件工程、软件可靠性、人工智能。E-mail: fu_bo@371.net

(责任编辑: 李铁柏)