文章编号:1673-0062(2018)02-0081-06

一种似然蜕变关系动态发现工具设计

范 超,阳小华,闫仕宇*,吴取劲,李 萌

(1.南华大学 计算机学院,湖南 衡阳 421001;2.中核集团高可信计算重点学科实验室,湖南 衡阳 421001)

摘 要:蜕变测试技术认为,测试中成功的测试用例可为构造蜕变关系提供有价值的信息,而似然蜕变关系的动态发现方法是根据已经成功运行的测试数据来发现蜕变关系的启发信息,基于一种似然蜕变关系发现算法的基本框架,进一步具体设计和实现算法,且开发相应的工具,实验表明该算法的可行性及工具的实用性.

关键词:蜕变测试;蜕变关系;似然蜕变关系;算法实现

中图分类号:TP311 **文献标志码**:A

Design of a Dynamic Discovery Tool for Likely Metamorphic Relation

FAN Chao, YANG Xiao-hua, YAN Shi-yu*, WU Qu-jing, LI Meng

(1.School of Computer, University of South China, Hengyang, Hunan 421001, China;
2.CNNC Key Laboratory on High Trusted Computing, Hengyang, Hunan 421001, China)

Abstract: The technology of metamorphic testing considers that the successful test cases can provide valuable information for constructing the metamorphic relation, and the likely metamorphic relations dynamic discovery method can discover the heuristic information of the metamorphic relation based on the test data that has been successfully operated. Based on a dynamic likely metamorphic relations discovery algorithm, this paper designs and implements the corresponding tools, and verifies the feasibility of the algorithm through an example.

key words: metamorphic testing; metamorphic relation; likely metamorphic relations; algorithm implementation

收稿日期:2018-01-24

基金项目:湖南省教育厅科学研究项目(16C1379);衡阳市科技计划项目(2017KJ273)

作者简介:范 超(1994-),女,硕士研究生,主要从事软件测试方向的研究.E-mail:731093728@qq.com. * 通讯作者:闫仕宇,E-mail:yanshiyu@usc.edu.cn

0 引 言

蜕变测试的概念是由 Chen 等人提出的,该技术是为了解决测试 oracle 问题[1].蜕变测试技术认为在测试的过程中,没有发现错误的测试用例(成功的测试用例)蕴含着有价值的信息,可用来构造新的测试用例对程序进行更加深入的检测[2].

蜕变测试技术已经被广泛的应用于数值型程序、图形学软件、并行编译器、图像处理程序和交互式软件等领域[3-8],其中蜕变关系是蜕变测试技术的关键.蜕变关系的发现方法有静态分析方法和动态发现方法,其中静态分析方法有 Chen 等人根据 sin 函数的性质定义了 10 条蜕变关系[4],求解偏微分方程程序根据聚焦属性构造出蜕变关系[4]等;动态发现方法有 Zhang 等人提出了基于搜索算法的多项式蜕变关系自动生成策略[9].Su 等人实现了一种能够根据同一方法的不同版本检测出不同的蜕变属性集的工具 KABU[10].其中借鉴动态不变量发现方法的思想,提出一种通用的似然蜕变关系的动态发现算法框架,为蜕变关系动态发现方法的深入研究提供新的思路,但该文献没有设计具体的程序和工具来实现该算法框架.因此本文基于似然蜕变关系发现方法的框架,进一步具体设计和实现算法,且开发相应的工具,实验表明该算法的可行性及工具的实用性.

1 基本概念

1.1 相关定义

定义 1[1] 假设程序 P 用来计算函数 $f(x)$,设 $x_1,x_2,\cdots,x_n,n>1$ 是 x 的任意 n 个变元值;若 $x_1,x_2,\cdots,x_n$ 之间满足关系 r 时,有 $f(x_1),f(x_2),\cdots,f(x_n)$ 满足关系 R,即:

$$r(x_1,x_2,\cdots,x_n)\rightarrow R(f(x_1),f(x_2),\cdots,f(x_n)) \tag{1}$$

则称 (r,R) 是 P 的 n 元蜕变关系,其中 r 称为蜕变关系的输入关系,R 称为输出关系.

由于程序 P 的输入和输出都是定义在有限的离散空间上,因此引入如下定义.

定义 2 假设 P 是用来计算函数 $f(x)$ 的程序,D 是 P 的输入域,若对于任意的 $x_1,x_2,\cdots,x_n\in D$,都有

$$r(x_1,x_2,\cdots,x_n)\rightarrow R(f(x_1),f(x_2),\cdots,f(x_n)) \tag{2}$$

则称 (r,R) 是 P 的关于 D 的蜕变关系,简称蜕变关系.

显然,如果 P 是正确的,那么对于任意的 $x_1,x_2,\cdots,x_n\in D$,都有

$$r(x_1,x_2,\cdots,x_n)\rightarrow R(P(x_1),P(x_2),\cdots,P(x_n)) \tag{3}$$

对似然蜕变关系进行了定义,定义如下:

定义 3 假设 P 的运行轨迹是 $T=\{(x_i,y_i)\mid x_i\in D,y_i=P(x_i),i=1,2,\cdots,N\}$,记 $T_x=\{x_i\mid(x_i,y_i)\in T\}$,若对于任意的 $x_{i1},x_{i2},\cdots,x_{in}\in T_x,n\ll N$,都有

$$r'(x_{i1},x_{i2},\cdots,x_{in})\rightarrow R'(y_{i1},y_{i2},\cdots,y_{in}) \tag{4}$$

则称 (r',R') 是 P 关于 T 的似然蜕变关系,简称似然蜕变关系.

显然,如果 P 是正确的,那么 P 关于 D 的蜕变关系一定是 P 关于 T 的似然蜕变关系,反之则不然.因此,可以通过发现 P 的似然蜕变关系来帮助构造 P 的蜕变关系.

1.2 算法思路

通过对似然蜕变关系的调查研究,提出了一种通用的似然蜕变关系的动态发现算法(discovery algorithm of likely metamorphic relations, DALMR)的框架,算法的核心思想是:

首先,设定预期能够发现的 r 关系形式库 Dr 和 R 关系形式库 DR;

其次,假设已有程序 P 的运行轨迹 T,则可以构造 n 元笛卡尔空间 $T^n x$;

再次,根据预设形式库 Dr,在 $T^n x$ 上搜索满足特定关系的子集,若子集中元素的个数足够多,则认为它是一个可能的蜕变关系输入 r';

最后,把满足关系 r' 的 n 元组 $(x'_1,x'_2,\cdots,x'_n)$ 对应的输出 $(y'_1,y'_2,\cdots,y'_n)$ 形成的 n 元组集合 $T^n yr'(T^n yr'=\{(P(x'_1),P(x'_2),\cdots,P(x'_n))\mid x'_1,x'_2,\cdots,x'_n\in r'\})$ 视为相应 R 关系的寻找空间,根据预设形式库 DR,寻找对应的蜕变关系输出 R',从而得到似然蜕变关系 (r',R').

本文基于似然蜕变关系的动态发现算法框架,设计并实现算法,开发相应的工具.

2 算法设计

算法的任务是从程序运行轨迹数据中发现蜕变关系的启发信息进而确定似然蜕变关系,要解决的问题包括如何预设关系形式库中的关系类

型、程序运行轨迹数据的以何种形式存放、如何确定关系形式中的参数等.

算法的主要步骤包括:1)数据预处理;2)取输入参数字段再做笛卡儿积;3)遍历输入关系形式库找出输入关系;4)根据输入元组映射出输出元组;5)遍历输出关系形式库找到输出关系;6)将找到的输入输出关系添加到似然蜕变关系集合.算法具体步骤如图 1 所示.

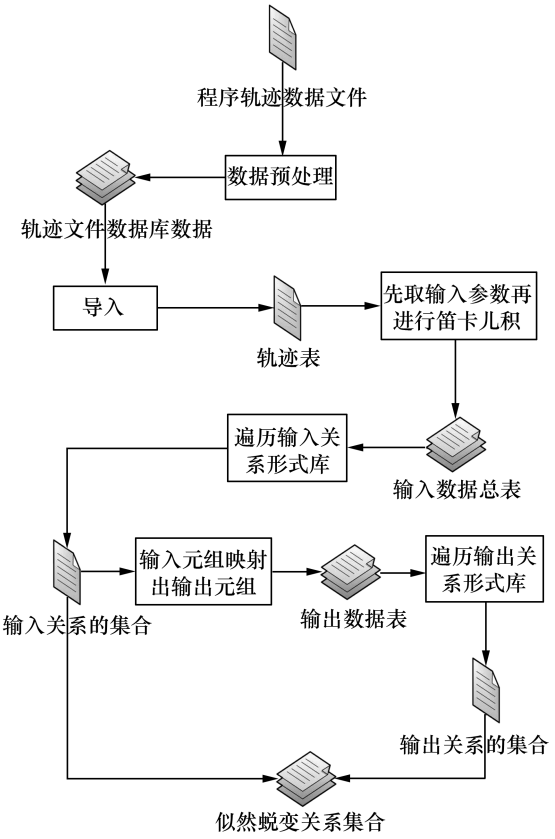

图 1　似然蜕变关系发现步骤

Fig.1　Likely metamorphic relation discovery step

2.1　预处理工作

数据预处理工作主要包括两个方面,首先是要将获得的运行轨迹数据文件进行格式转换,得到轨迹文件数据库数据,并将轨迹数据文件存储到 MYSQL 数据库中.假设计算函数程序中的输入输出是二元关系,则存储到 MYSQL 数据库表中的字段包含 ID 字段(测试用例编号)、X 字段(输入参数字段)、Y 字段(输出结果字段).再是要将预设的关系形式存储到数据库中,预设的关系形式主要包括线性关系、比较关系、函数关系等,使得用户能够根据自己的需要自由地选择关系形式,进一步增强算法的灵活性.

2.2　数据处理

1)找输入关系

首先,将轨迹数据文件中的 X 字段与其自身进行笛卡儿积,即 $T^2x=D(xi)\times D(xj)$,再依据关系形式和经验值 K 来求取 T^2x 空间上的输入关系集合 Dr′;

```
Dr′=empty;
//将 T²x 空间上找到的输入关系集合 Dr′置空
for a from Amin to Amax
//通过参数 a 的范围枚举 a
for b from Bmin to Bmax
//通过参数 b 的范围枚举 b
if(gcd(a,b)==1 and a! =0 and b! =0 and a! =b)        //参数 a 和 b 互质
    Dr.push(a*xi+b*xj)
//以线性关系为例,根据 a 和 b 生成二元表达形式存入关系形式库 Dr
end if
end for
end for
while  (Dr≠Ø)
r=dequeue(Dr);
//从输入关系形式库 Dr 中选择未处理过的形式 r
for(T²x 每个元组)
select M=count(C1) from T²x where a*xi+b*xj=C1;
//统计 a*xi+b*xj=C1 中计算结果 C1 的个数 M
if(M≥K)        //根据经验值 K 取 10
then
确定关系 r′为 a*xi+b*xj=C1 等;
//实际情况中可能存在多个
把 r′添加到 Dr′;
记录满足 r′的有序对;
//设有序对的空间为 T²xr′;
else
break;
end if
end for
end while
```

2)找似然蜕变关系

依据 T^2xr'空间中 r'映射出对应的输出 T^2yr'空间,并在此映射空间上求取关系 R',将最终得到的$\{r'\rightarrow R'\}$添加到似然蜕变关系集合.

```
DR′=empty;
//输出关系集合 DR′初值置空;
while(Dr′≠Ø)
r′=dequeue(Dr′);
//选取输入关系集合 Dr′中的输入关系 r′;
由 T²xr′向 T²y 投影,截取子空间 T²yr′;
//T²yr′={(y_i,y_j)|(x_i,x_j)属于 r′)
for a from Amin to Amax
//通过参数 a 的范围枚举 a
for b from Bmin to Bmax
//通过参数 b 的范围枚举 b
if(gcd(a,b)==1 and a!=0 and b!=0 and a!=b)        //参数 a 和 b 互质
    Dr.push(a*yi+b*yj)
//以线性关系为例,根据 a 和 b 生成二元表达形式存入关系形式库 DR
end if
end for
end for
while(DR≠Ø)
R=dequeue(DR)
//从输出关系形式库 DR 中选择未处理过的形式 R;
for(T²yr′每个元组)
select N=count(C2) from T²x where a*yi+b*yj=C2;
//统计 a*yi+b*yj=C2 中计算结果 C2 的个数 N
if (M==N)        //满足某输入关系的有序对个数是否等于满足某输出关系的有序对个数
then
确定关系 R′为 a*yi+b*yj=C2 等;
//实际情况中可能存在多个
把 R′添加到 DR′;
{r′→R′}添加到 LMRs;
else
break;
end if
end for
end while
end while
```

2.3 工具实现

似然蜕变关系发现工具主要的功能包括:导入程序轨迹数据文件、找出输入关系集合、找出似然蜕变关系集合,具体功能如图 2 所示.

根据工具实现的主要功能,采取面向对象 JAVA 语言进行程序设计开发,最终得到工具的总界面如图 3 所示.

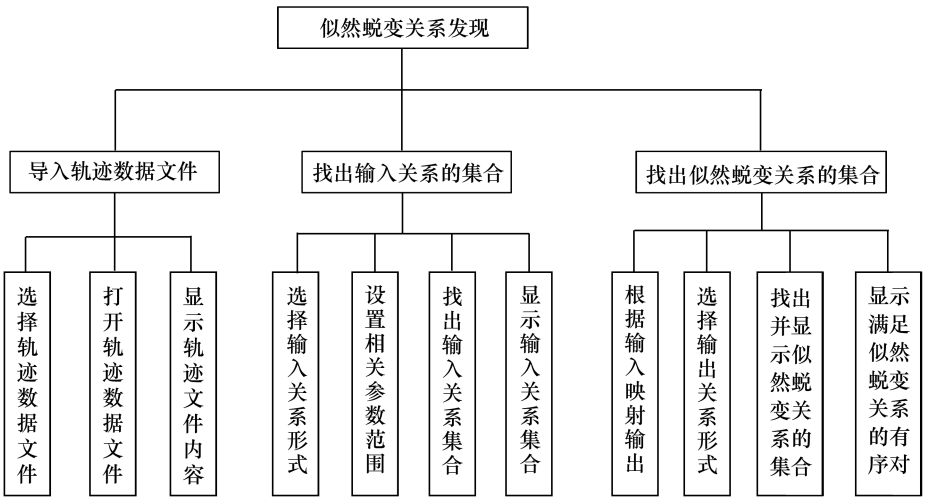

图 2 似然蜕变关系发现工具功能图

Fig.2 Likely metamorphic relation discovery tool function diagram

图 3　似然蜕变关系发现工具界面

Fig.3　Likely metamorphic relation discovery tool interface

3　数值实验

本文选择简单的计算函数程序 P 作为实验的待测程序,使用似然蜕变关系发现工具进行似然蜕变关系的发现.

实验环境:Intel(R)Corei5-4210MCPU2.60GHz

实验平台:似然蜕变关系发现工具

1)实验程序:选取计算正弦函数的程序 $P1$、计算余弦函数的程序 $P2$、计算正切函数的程序 $P3$、计算指数函数的程序 $P4$、计算对数函数的程序 $P5$ 作为实验对象.

2)实验数据:分别取已有的 100 个、200 个、300 个、400 个测试用例作为四组计算函数程序的轨迹数据进行实验.

3)运行结果:将轨迹数据在似然蜕变关系发现工具上运行并得出实验结果,在这四组测试用例中都可以找到如表 1 所示的实验结果,且实验中不同的测试用例个数在执行过程中所用的时间如表 2 所示.为了直观的展示执行时间随测试用例数增多的变化情况,依据表 2 绘制图 4.

4)结果分析:由表 1 表明,该工具找出了实验程序的似然蜕变关系,比如,$P1$ 的三个似然蜕变关系,其中 $R1\{Xi-Xj=2\pi\rightarrow Yi-Yj=0\}$ 跟计算函数的性质进行比较,发现 $R1$ 符合计算函数性质推导出来的蜕变关系,说明该工具对于蜕变关系的发现具有较高的可信度.从表 2、图 4 中可看出,随着测试用例数(轨迹数据)的增多,数据存储时间以及笛卡儿积计算所消耗的时间也逐渐增长,因此算法的执行效率降低.为了保证算法的执行效率,需对算法的时间和空间复杂度进一步优化.

表 1　实验结果

Table1　Experimental results

实验程序	似然蜕变关系集合
$P1$	$R1:\{Xi-Xj=2\pi\rightarrow Yi-Yj=0\}$ $R2:\{Xi-Xj=\pi\rightarrow Yi+Yj=0\}$ $R3:\{Xi+Xj=0\rightarrow Yi+Yj=0\}$
$P2$	$R1:\{Xi-Xj=2\pi\rightarrow Yi-Yj=0\}$ $R2:\{Xi-Xj=\pi\rightarrow Yi+Yj=0\}$ $R3:\{Xi+Xj=0\rightarrow Yi-Yj=0\}$
$P3$	$R1:\{Xi+Xj=\pi\rightarrow Yi+Yj=0\}$ $R2:\{Xi-Xj=\pi\rightarrow Yi-Yj=0\}$ $R3:\{Xi+Xj=0\rightarrow Yi+Yj=0\}$
$P4$	$R1:\{Xi+Xj=0\rightarrow Yi*Yj=1\}$
$P5$	$R1:\{Xi*Xj=1\rightarrow Yi+Yj=0\}$

表 2　不同的用例数在各操作阶段所用时间

Table 2　Time used for different use cases at each operation stage

数据	100 个用例/ms	200 个用例/ms	300 个用例/ms	400 个用例/ms
导入轨迹数据	260	306	440	472
笛卡儿积操作	0.401	0.467	0.853	2.282
输入关系计算	0.495	1.441	5.786	8.410

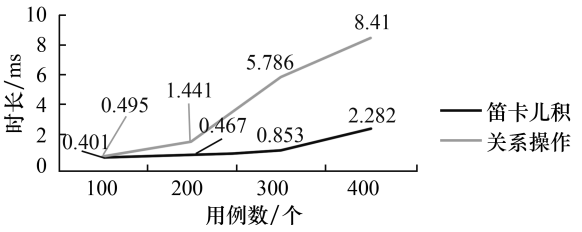

图 4 计算时长随用例数增多的变化情况

Fig.4 The change of calculation time with increase in the number of cases

4 结 论

根据似然蜕变关系动态发现方法设计的算法及实现的似然蜕变关系发现工具能够依据测试数据找出计算函数程序的蜕变关系,而对于复杂的计算函数程序,该工具虽能找到似然蜕变关系,但其中可能存在伪似然蜕变关系.因此下一步工作要增加程序轨迹数据个数、扩充预设的关系形式库,为真正蜕变关系的构造提供有效的启发信息.

参考文献:

[1] CHEN L, CAI L, LIU J, et al. A generating criterion of test cases based on equivalence classes for meta-morphic testing[J]. International journal of digital content technology & its applications, 2013, 7(5): 1184-1193.

[2] 董国伟,徐宝文,陈林,等.蜕变测试技术综述[J].计算机科学与探索,2009,3(2):130-143.

[3] ZHOU Z Q, HUANG D H, TSE T H, et al. Metamorphic testing and its applications[C]//Proceedings of the 8th International Symposium on Future Software Technology. Tokyo: Software Engineers Association, 2004: 1-5.

[4] CHEN T Y, FENG J, TSE T H. Metamorphic testing of programs on partial differential equations: a case study [C]//Proceedings 26th Annual International Computer Software and Applications. Oxford, England: IEEE Computer Society, 2002: 327-327.

[5] MAYER J, GUDERLEI R. On random testing of image processing applications [C]//International Conference on Quality Software. Czech Republic: IEEE Computer Society, 2006: 85-92.

[6] CHAN W K, HO J C F, TSE T H. Piping classification to metamorphic testing: an empirical study towards better effectiveness for the identification of failures in mesh simplification programs [C]//International Computer Software and Applications Conference. Beijing: IEEE, 2007: 397-404.

[7] SIM K Y, PAO W K S, LIN C. Metamorphic testing using geometric interrogation technique and its applications [J]. Ecti transactions on computer and information technology, 2006, 1(2): 91-95.

[8] CHAN W K, CHEUNG S C, LEUNG K P H. A metamorphic testing approach for on-line testing of service-oriented software applications[J]. International journal of web services research, 2007, 4(2): 61-81.

[9] ZHANG J, CHEN D, Hao D, et al. Search-based Inference of Polynomial metamorphic relations[C]//Proceedings of the 29th ACM/IEEE international conference on Automated software engineering. New York: ACM, 2014: 701-712.

[10] SU F H, BELL J, MURPHY C, et al. Dynamic Inference of Likely Metamorphic Properties to Support Differential Testing[C]//IEEE/ACM, International Workshop on Automation of Software Test. Austin, TX (USA): IEEE, 2015: 55-59.

(责任编辑:扶文静)

(上接第 80 页)

[16] YANG C, ZHANG L, LU H, et al. Saliency detection via graph-based manifold ranking[C]// IEEE conference on computer vision and pattern recognition. IEEE computer society, 2013: 3166-3173.

[17] CHEN J, LI Z, HUANG B. Linear spectral clustering superpixel. [J]. IEEE transactions on image processing, 2017, 26(7): 3317-3330.

[18] ACHANTA R, SHAJI A, SMITH K, et al. SLIC superpixels compared to state-of-the-art superpixel methods. [J]. IEEE transactions on pattern analysis & machine intelligence, 2012, 34(11): 2274-2282.

[19] RAO A V, MEASE K D. Eigenvector approximate dichotomic basis method for solving hyper-sensitive optimal control problems [J]. Optimal control applications & methods, 2015, 21(1): 1-19.

[20] JIANG B, ZHANG L, LU H, et al. Saliency detection via absorbing markov chain[C]// IEEE international conference on computer vision. IEEE, 2014: 1665-1672.

[21] 曾玖贞,王超,王彦,等.基于卷积神经网络的钢轨测量廓形畸变动态识别[J].南华大学学报(自然科学版),2017,31(1):47-53.

(责任编辑:龙威)