文章编号:1673 - 0062(2012)01 - 0063 - 05

基于 GEP 的线性指数函数型程序不变量动态发现方法

阳小华,黄彩霞

(南华大学 计算机科学与技术学院,湖南 衡阳 421001)

摘 要:不变量是用来描述程序运行时保持不变性质的逻辑断言.根据关系数据理论,程序不变量可分函数依赖型和非函数依赖型程序不变量.着眼于函数依赖型程序不变量,借助 GEP 的函数发现特点和 Daikon 对线性程序不变量的发现能力,重点对线性指数函数型程序不变量动态发现方法进行研究,通过实验证明了 GEP 对线性指数形式的函数有较高的发现效率,可以扩展 Daikon 在线性指数函数型程序不变量方面的预置形式以达到从程序轨迹数据中发现该类程序不变量的目的.

关键词:GEP;程序运行轨迹数据;启发性信息;线性指数函数形式;程序不变量;不变量动态发现

中图分类号:TP311 **文献标识码**:A

Methods on Lineal and Exponential Function Program Invariants Dynamically Mining Based on GEP

YANG Xiao-hua, HUANG Cai-xia

(School of Computer Science and Technology, University of South China, Hengyang, Hunan 421001, China)

Abstract: Invariants is used to describe logical assertions of which running programs keep unchanged. According to the relational data theory, program invariants can be divided into Function Program Invariants and Non-Function Program Invariants. The thesis beginning from function program invariants focuses on the research on Lineal and Exponential Function Invariants Dynamically Mining with the help of GEP feature of Function Mining and Daikon ability to find out the Lineal and Exponential Function Program Invariants. The experiment proves the high efficiency of GEP discovering Lineal and Exponential function type, which expand the type Daikon preset for Function Program Invariants in order to find more invariants of this type from Program-Running track data.

key words: GEP; program-running track data; heuristic information; lineal and exponential

收稿日期:2012 - 02 - 09

基金项目:湖南省自然科学基金资助项目(10JJ6097);湖南省高等学校科学研究重点基金资助项目(11A105)

作者简介:阳小华(1963 -),男,湖南衡阳人,南华大学计算机科学与技术学院教授,博士生导师.主要研究方向:知识科学与信息检索.

function type;program invariants;invariants dynamically mining1

0 引 言

程序中蕴含的不变性质即程序不变量,它用来描述程序的运行规律. 不变量的发现主要有静态分析和动态发现两种方法. 静态分析主要是通过检查程序代码等文档以发现其中蕴含的不变性质,这类技术通常只能应用于中小规模的程序. 动态发现检测的是程序多次运行中保持不变的性质,是在程序运行过程中通过获得程序轨迹数据并对其分析得到的程序不变量,适用于较大规模程序.

目前,动态程序不变量研究方面影响最大的是美国 MIT 程序分析组 Ernst 等人开发的动态发现似然程序不变量工具 Daikon. Daikon 的核心思想是预置一个不变量测试库,对测试库中的每个不变量,用轨迹数据进行检测,根据可信度对满足条件的不变量进行统计验证,并报告不变量[1]. 截至 2007 年,Daikon 工具预置了 75 种不变量形式来构成不变量测试库[2]. 它可以很好的发现程序中蕴含的不变量,但同时也存在着很大的问题——不在预置形式中的不变量是不被检测出来的. 对函数形式的程序不变量来说,Daikon 只能检测出三元线性关系的不变量,并未涉及其它函数型不变量.

因此,如何丰富 Daikon 可检测的程序不变量,尤其是函数依赖型程序不变量,将是动态不变量发现的研究重点(Daikon 在设计上留有预置接口,以方便后续改进和扩展可检测的不变量形式). 针对函数型程序不变量,本文对线性指数函数型程序不变量的动态发现进行研究.

1 相关工作

在 Daikon 的影响下,斯坦福大学的 Hangal 博士等人开发了一个动态不变量发现与检测引擎 Diduce. Diduce 采用动态检测的方法进行不变量的检测,在程序的观察点插入假定不变量(包括函数依赖型程序不变量),并在程序运行过程中持续检查该点的行为是否违反了这些假定不变量,并报告出所有检测到的违例[3]. 虽然它的检测效率很高,但不变量的获取首先是来自于检测者的先验知识,主观能动性太强,可检测到的不变量数目较为有限.

针对 Daikon 盲目性和效率低下的特点,文献[4]利用函数关系存在性判定依据提出了启发式试探法,从而将程序不变量划分为函数型和非函数型程序不变量. 该方法通过引入关系数据理论对程序轨迹数据进行函数依赖推导,以发现蕴含的函数依赖集,集合非空,转入函数型不变量检测,否则则进行非函数型程序不变量的推导.

基因表达式编程 GEP(Gene Expression Programming)算法在不需要预知函数类型的情况下,以基本变量和算符为遗传物质,通过遗传、变异,可以发现多元函数,分段函数,和其它不能用解析式表达的关联规则,聚类等规则[5]. 同组研究者吴取劲同志在文献[6]中讨论了通过程序轨迹获取函数型程序不变量的技术,并提出基于 GEP 的函数依赖型程序不变量发现方法:1)利用关系数据理论判断函数型程序不变量的存在,并获取函数依赖集. 2)借助 GEP 的函数发现能力,在函数依赖集上运行测试用例,生成表达式. 3)依据步骤 2 获得启发信息,寻找相应的数据理论指导函数表达式的确定. 在该方法的指导下,作者通过实验得出:借助数学理论指导,GEP 技术的引入可以高效、准确的解决标准多项式形式程序不变量检测问题;而对于其它形式的多项式,则还需要进一步的研究. 他的工作对目前程序不变量发现技术来说是一重大的补充,增加了从程序轨迹数据中发现更多程序不变量的可能性. 此外,则该方法的提出为 Daikon 预置程序不变量的扩展及后续函数型程序不变量的检测提供了新的指导思路,加快了函数依赖型程序不变量检研究进程.

2 基于 GEP 的线性指数函数型程序不变量动态发现

GEP 算法在进化的过程中始终都是从函数符号集合和终结符集合中选取一定数量的元素组合生成染色体个体,依据适应度算法来确定函数的拟合程度,从而达到函数发现的目的[7]. 基于 GEP 的函数型程序不变量检测方法分为以下几个阶段:1)源程序编配;2)在测试用例集上运行编配程序,生成轨迹文件[4];3)对轨迹数据进行函数依赖分析,找出函数依赖集[6,8];4)在函数依赖集上利用 GEP 算法生成启发式信息[8];5)寻找相应的数学理论或技术生成似然程序不变量;6)检验并报告程序不变量.

对于蕴含某种函数关系的程序轨迹数据,如

果 GEP 算法可以很好的发现这种函数形式,那么该类不变量的发现问题将转换为多元线性程序不变量的发现. 由于 Daikon 可以检测三元线性程序不变量,借助于 Daikon 在设计时预留的接口,可以将这种函数形式预置到 Daikon 中,由 Daikon 来进行精确计算,确定函数表达式,完成相应的程序不变量的发现工作. 基于这种思路,本文对线性指数函数型程序不变量(即 y = k * exp(ax + b)这种形式的程序不变量)的发现进行了实验研究,研究的重点从线性指数函数型程序不变量的发现转移到了 GEP 对线性指数函数形式的发现能力上. 在指数函数计算中,k * exp(ax + b)可等价为 exp(ax + c),exp(ax)/d 等函数(a,b,c,d,k 为有理数),因此本文的线性指数函数专指指数函数 y = k * exp(ax + b)及它的等价函数,以下均用 y = k * exp(ax + b)(a,b,k 为有理数)来特指线性指数函数型程序不变量.

对不同形式的函数型程序不变量来说,为加快收敛速度,提高 GEP 函数发现能力,它在进化过程中必须同时满足两个条件:1)函数符号集合的元素必须是完备的;2)选择的适应度算法要满足特定需求(不同的适应度算法对 GEP 函数发现效率也是不同的). 由于线性指数函数的函数符号集是定的,因此需要对适应度算法进行实验分析.

2.1　适应度的选择

本次实验环境为:Intel(R)4CPU,1.80GHz;实验平台:APS 3.0;

实验函数:y = 5 * exp(2.3 * x + 1.7);训练样本数为 500,测试样本数为 4000,取值在[- 10,10]以内的随机数,部分 GEP 参数设置如表 1 所示.

表 1　GEP 参数设置

Table 1　GEP parameters setting

初始染色体个数	头部长度	基因个数	进化中止条件	变异概率	交叉概率
80	15	1	适应度 > 999	0.044	0.1

APS 中内置的适应度算法共 11 种,考虑到时间效率,本次实验选取收敛速度较快的六种进行实验研究,为了保持数据的稳定性,每组算法运行 20 次,实验结果如表 2 所示.

表 2　GEP 适度度效率比较

Table 2　Comparison to efficiency of GEP fitness

	平均遗传代数/s	80% 遗传代数/s	准确率(/20)
R-Square	198	156.2	18
MSE	455.32	420.8	17
RMSE	330.75	321.4	16
RSE	288	247.3	18
RRSE	487	444.6	17
RAE	251.6	227	19

注:在 GEP 进化过程中,遗传代数与收敛时间几乎成正比,因此,此处使用遗传代数作为进化速度的度量.

通过上图对比分析可知,R-Square 在以上适应度算法中进化速度最快,准确率次高;RAE 准确率最高,进化速度较前者则非常低. 其它几种适应度算法无论是在时间还是在准确率上都不如以上两种. 兼顾到 GEP 收敛速度和发现效率,线性指数函数依赖型不变量的研究选择 R-Square 作为本次实验的适应度函数.

2.2　基于 GEP 的线性指数函数型程序不变量发现实验

实验环境及参数设置同上;GEP 适应度算法选择 R-Square

实验方法:获取以下实验函数的轨迹点(x,y)作为程序轨迹的测试数据,分别在 APS 上运行 GEP 算法 100 次,记录相应的生成表达式并分析其形式,最后统计分析结果.

本文选择了以下几种不同形式的指数函数作为实验函数:

F1:y = exp(x)

F2:y = exp(x)/4

F3:y = exp(x + 1)

F4:y = 3.1 * exp(2x + 5)

F5:y = 3 * x * exp(x)

F6:y = exp(x^5 + x)

F7:y = x^3 + x^2

F8:y = exp(x) + x^3 + 1

实验结果统计如表 3 所示.

通过以上实验数据分析对比可知,当轨迹数据中含有线性指数关系时,GEP 可以很好的发现轨迹数据中蕴含的函数形式,发现效率可高达 100%;当轨迹数据含有非线性指数关系时,GEP 生成的函数形式多样化,即使经过简化或转换,GEP 在形式发现效率上依旧较低;当轨迹数据中不含指数项时,

GEP 生成的表达式含指数项的概率较小.

表 3 GEP 实验结果统计

Table 3 GEP statistics of experiment result

实验组	生成表达式形式	出现次数/(实验次数(100))
F1	Y = exp(x)	100
F2	Y = exp(x + a)及其等价类	100
F3	同 F2	100
F4	同 F2	100
F5	Y = a * x * exp(x) + b	81
	Y = a * x * exp(x) + b * x + c	26
	Y = a * x * exp(x) + b * x * x	3
F6	Y = exp(x^a + x * b)	38
	Y = exp(x^a + x * x * b + x * c)	12
	Y = exp(x^a) + exp^(x * d)	23
	Y = exp(x^a + x * x * b) + x * d	37
F7	Y = x^a + x^b	87
	Y = x^a + x^b + x * c + d	13
F8	Y = exp(x) + x^a + b	95
	Y = exp(x) + exp(exp(x * a) + b)	5

注:以上函数形式中,a,b,c 表示常数.

从上述实验可以看出,GEP 算法对线性指数函数的发现效果较好. 为了验证该论断,本次实验选取了 2 个函数,随机生成 100 组测试用例,每组测试用例 2500 个样本数据. 将各组测试用例置入 ASP 中进行遗传变异操作,遗传条件同上,记录可信度为 1000 时的生成表达式并分析其形式,最后统计分析结果. 实验函数如下:

F1:y = (exp(2 * x - 1/7))/3.1

F2:y = 2 * exp(6.7 * x + 6.15)

实验结果如图 1 所示.

以上实验结果显示:GEP 算法有能力从程序运行轨迹找出蕴含 y = k * exp(ax + b)性质的线性指数函数,发现效率很高,但由于 GEP 算法在进化过程中有时会陷入"局部最优"从而导致函数收敛速度减慢,再加上计算偏差,最后进化而来的生成表达式可能与原函数形式上并不一致. 然而,从整体上来说,GEP 算法对线性指数函数依然是有着极高的发现效率.

这一结论对程序不变量的发现有着极其重要的作用. 从轨迹数据中发现程序不变量时,通过 GEP 算法如果能够获得线性指数函数形式的启发式信息,那么就可以将该形式预置到 Daikon 中,借助 Daikon 能够发现三元线性函数的特点,由 Daikon 完成线性指数函数型程序不变量的发现.

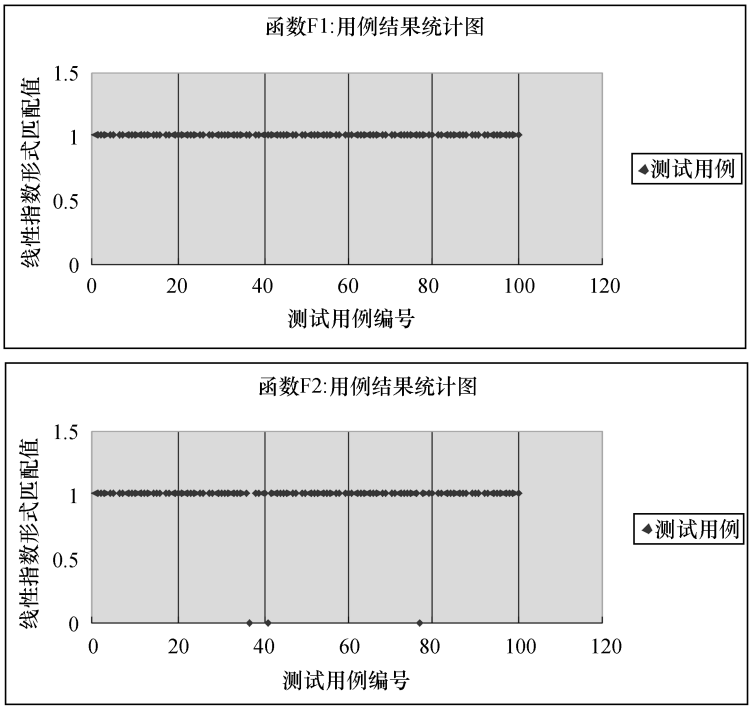

图 1 测试用例结果统计图

Fig. 1 Test case results chart

3　结束语

线性指数函数在实际中有重要的应用价值,比如在可靠性工程中,指数概率分布函数 $y = k * \exp(-kx)$ 就广泛用来表示元件失效率. 本文通过对函数型程序不变量发现方法的分析,着重讨论基于 GEP 的线性指数函数型程序不变量发现方法并进行实验,从而解决了 $y = k * \exp(ax + b)$ (a,b,k 为有理数)形式的程序不变量发现问题. 它扩展了 Daikon 的预置形式,增加了从程序轨迹中发现更多程序不变量的可能性.

下一步的工作:用 GEP 对其它类型的指数函数型不变量开展研究,特别是一些常见的概率分布函数,如正态分布函数等等.

参考文献:

[1] Michael D Ernst, Jak Cockrell, William G Griswold, et al. Dynamically discovering likely program to support program evolution[J]. IEEE Transactions on Software Engineering, 2001, 27(2): 99-123.

[2] Michael D Ernst. Dynamically discovering likely program invariants[D]. Washington: University of Washington Department of Computer Science and Engineering, 2000.

[3] Hangal S, Lam M S. Tracking down software bugs using automatic anomaly detection[C]//Proceedings of the 24th International Conference on Software Engineering, 2002: 291-301.

[4] 刘杰. 函数依赖似然不变量发现方法研究[J]. 研究与开发, 2008, 28(3): 33-35.

[5] 唐常杰,张天庆,左劫,等. 基于基因表达式编程的知识发现[J]. 计算机应用, 2004, 24(10): 7-10.

[6] 吴取劲. 基于 GEP 的函数型似然程序不变量发现方法[D]. 衡阳:南华大学, 2009.

[7] 黄晓冬,唐常杰,李智,等. 基于基因表达式编程的函数关系发现方法[C]//中国计算机学会,第二十届全国数据库学术会议. 长沙, 2003: 278-282.

[8] 刘树锟,阳小华. 一种函数依赖程序不变量动态检测方法[J]. 微电子学与计算机, 2008, 25(7): 205-209.