基于 LASSO-SVM 的软件缺陷预测模型研究*

吴晓萍[1], 赵学靖[1], 乔 辉[2], 刘东梅[1], 王 志[1]
(1. 兰州大学 应用数学与统计学院, 兰州 730000; 2. 中国人民解放军信息工程大学 密码工程学院, 郑州 450004)

摘 要: 针对当前大多数软件缺陷预测模型预测准确率较差的问题,提出了结合最小绝对值压缩和选择方法与支持向量机算法的软件缺陷预测模型。首先利用最小绝对值压缩与选择方法的特征选择能力降低了原始数据集的维度,去除了与软件缺陷预测不相关的数据集;然后利用交叉验证算法的参数寻优能力找到支持向量机的最优相关参数;最后运用支持向量机的非线性运算能力完成了软件缺陷预测。仿真实验结果表明,所提出的缺陷预测模型与传统的缺陷预测模型相比具有较高的预测准确率,且预测速度更快。

关键词: 软件缺陷预测; 最小绝对值压缩与选择方法; 特征选择; 支持向量机; 交叉验证
中图分类号: TP311.5 **文献标志码**: A **文章编号**: 1001-3695(2013)09-2748-04
doi:10.3969/j.issn.1001-3695.2013.09.047

Research of software defect prediction model based on LASSO-SVM

WU Xiao-ping[1], ZHAO Xue-jing[1], QIAO Hui[2], LIU Dong-mei[1], WANG Zhi[1]
(1. *School of Statistics & Mathematics*, *Lanzhou University*, *Lanzhou* 730000, *China*; 2. *Institute of Cryptography Engineering*, *PLA Information Engineering University*, *Zhengzhou* 450004, *China*)

Abstract: The prediction accuracy of most current software defect prediction models is not very high. To solve the problem, this paper investigated a software defect prediction model with the least absolute shrinkage and select operator (LASSO) and the support vector machine (SVM). At first, it reduced the dimension of the original data sets and extracted the data which was irrelevant with software defect prediction by taking advantage of the feature selection capability of LASSO. Then it found the correlated optimal weights of the SVM by making use of the parameter preference capability of cross validation. At last, it finished software defect prediction by utilizing the non-linear computing power of SVM. The simulation experiment indicated that proposed method owe a higher prediction precision than the traditional ones and predicted faster.

Key words: software defect prediction; least absolute shrinkage and select operator (LASSO); feature selection; support vector machine (SVM); cross validation

0 引言

目前统计方法、机器学习和混合统计与机器学习三种方法已经被广泛地应用于软件缺陷预测领域。在这三种方法中,混合统计与机器学习方法已经被证明可以更好地实现软件缺陷预测[1]。

在软件缺陷预测领域,由于软件模块的复杂性、软件复杂性度量和易出错模块类别的对应关系很难确定,研究人员提出了采用大量的原始复杂性度量属性数据来进行软件缺陷预测。然而不加选择地引入所有的复杂性度量属性来进行缺陷预测,往往不能取得良好的效果[2]。这是因为过多地引入复杂性度量属性会导致“维数灾难”和“组合爆炸”[3],使得计算量空前增大,估计和预测精度也会下降;此外,在一些情况下,获得某些软件复杂性度量属性数据代价昂贵,如果这些数据本身对预测结果的影响微乎其微,那么势必会造成缺陷数据的收集和模型应用的费用不必要地增大。

针对上述问题,混合统计与机器学习方法的核心是先利用统计方法的思想对要进行缺陷预测的数据集进行特征选择[4,5]处理,达到降低数据集维数的目的;之后用机器学习方法对处理后的数据集进行缺陷预测,获得所需要的预测精度。

本文基于上述思想,提出了一种基于 LASSO-SVM 的软件缺陷预测方法。其主要工作如下:分析了 LASSO 方法的核心思想,以及如何使用它的特征选择能力来得到精简的复杂性度量属性子集;分析了 SVM 机器学习算法的核心思想,并且使用交叉验证算法对 SVM 的相关参数进行优化,获得 SVM 优化模型;将使用 LASSO 方法得到的属性精简子集输入到基于 SVM 机器学习算法的软件缺陷预测模型中,并与其他传统软件缺陷预测方法进行对比,验证了该方法的有效性和预测准确率。

1 相关研究

1.1 LASSO 方法

Tibshirani 于 1996 年提出了一种有偏估计的方法,被用来处理高维数据的特征选择问题,该方法称为最小绝对值压缩与选择方法(LASSO)[6]。LASSO 可以通过惩罚函数将影响较小的观察变量的回归系数收缩到 0,这样做虽然牺牲了一定的估

收稿日期: 2012-12-07; **修回日期**: 2013-02-24 **基金项目**: 国家教育部留学回国人员科研启动基金资助项目(第 44 批);兰州大学中央高校基本科研业务费专项资金资助项目(lzujbky-2012-15,lzujbky-2013-178)

作者简介: 吴晓萍(1987-),女,陕西渭南人,硕士研究生,主要研究方向为金融软件系统建模(353054349@qq.com);赵学靖(1972-),男,甘肃临洮人,副教授,博士,主要研究方向为高维数据统计分析;乔辉(1988-),男,陕西渭南人,硕士研究生,主要研究方向为软件可靠性预测;刘东梅(1986-),女,甘肃靖远人,硕士研究生,主要研究方向为生物统计;王志(1987-),男,湖南长沙人,硕士研究生,主要研究方向为经济统计.

计偏差,但能降低预测的方差从而提高预测的精确性。

使用 LASSO 方法进行软件缺陷预测时首先需要对复杂性度量属性数据集进行中心标准化处理,以消除不同指标量纲的影响,使其满足

$$\sum_{i=1}^{n} y_i = 0, \sum_{i=1}^{n} x_{ij} = 0, \sum_{i=1}^{n} x_{ij}^2 = 1 \quad j = 1, \cdots, m \tag{1}$$

其中:m 为维数,n 为变量数。具体到软件缺陷预测中,m 为软件复杂性度量数据集中各模块的属性数目,如可将 HALSTEAD 度量[7]、McCabe 度量[8] 和 C&K[9] 度量等作为维数,而 n 则为数据集中的模块数。假设一个数据集由 458 个模块数组成,每个模块均拥有 42 个复杂性度量属性,则 $m = 42, n = 458$。

$$\min S(\,)x_{j\beta} \tag{2}$$

s. t.

$$T(\hat{\beta}) = \sum_{j=1}^{m} |\hat{\beta}_j| \leqslant t \quad T(\hat{\beta}) = \sum_{j=1}^{m} |\hat{\beta}_j| \leqslant t \tag{3}$$

其中:

$$S(\hat{\beta}) = \| Y - \hat{\mu} \|^2 = \sum_{i=1}^{n} (y_i - \hat{\mu}_i)^2, \hat{\mu} = X\hat{\beta} = x_{i,j}\hat{\beta}_j$$

y_i 是响应变量;$x_i = (x_{i1}, x_{i2}, \cdots, x_{in})$ 是观察变量;$\hat{\beta}_j$ 为第 j 个变量的回归系数。

LASSO 方法,即为在 $T(\hat{\beta}) \leqslant t$ 的约束条件下,通过调整各变量的回归系数,即 $\hat{\beta}_j$ 的值来求响应变量 y_i 与回归变量 $\hat{\mu}$ 的最小平方差和,即求 $S(\hat{\beta})$ 的最小值。

LASSO 方法在基于分类的软件缺陷预测模型中,y_i 为 0 ~ 1 属性变量,用来反映数据集中的模块是否为有缺陷模型,而 x_i 则为其他的复杂性度量属性变量。

通过对回归系数添加约束条件式(3),可以将与响应变量相关度较低的观察变量的回归系数 $\hat{\beta}_j$ 置为 0,从而在特征选择时将这些回归系数为 0 的变量去除,只保留强相关性的变量。t 的取值可以为 0 ~ ∞。当 t 的值比较小时,某些相关度低的变量系数就被压缩为 0,从而将这些变量删除,以达到特征选择的目的;当 t 取得足够大时,将不再具有约束的作用,此时所有的属性都将被选择并且形成一个变量选择序列。

图 1 为 LASSO 对模拟数据的变量选择序列分析[10]。

1.2　支持向量机

支持向量机(support vector machine,SVM)[11] 是由 Vapnik 等人首先提出的,其已经广泛应用于模式识别和非线性回归领域。支持向量机的主要思想是建立一个分类超平面作为决策曲面,使得正例与反例之间的隔离边缘被最大化。支持向量机的理论基础是统计学习理论,更精确地说,支持向量机是结构风险最小化的近似实现。该原理基于这样的事实:学习机器在测试数据上的误差率(即泛化误差率)以训练误差率和一个依赖于 VC 维数(Vapnik-Chervonenkis dimension)的项的和为界,在可分模式情况下,支持向量机对于前一项的值为零,并且使第二项最小化。因此,尽管它不利用问题的领域内部问题,但在模式分类问题上支持向量机能提供好的泛化性能,该属性是支持向量机所特有的。

支持向量机的体系结构如图 2 所示。其中,K 为核函数种类主要有:

a)线性核函数。

$$K(x, x_i) = x^{\mathrm{T}} x_i$$

b)多项式核函数。

$$K(x, x_i) = (\gamma x^{\mathrm{T}} x_i + r)^p \quad \gamma > 0$$

c)径向基核函数。

$$K(x, x_i) = \exp(-\gamma \| x - x_i \|^2) \quad \gamma > 0$$

d)两层感知器核函数。

$$K(x, x_i) = \tanh(\gamma x^{\mathrm{T}} x_i + r) \quad \gamma > 0$$

在上述的核函数中,径向基核函数(radial basis function,RBF)具有较宽的收敛范围,是较理想的分类依据函数。本文支持向量机分类算法中采用的正是 RBF 核函数。

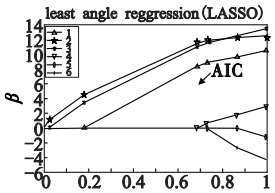

图1　LASSO对模拟数据的特征序列

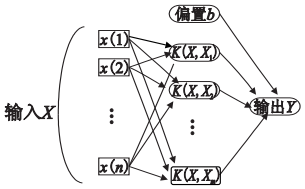

图2　支持向量机的体系结构

1.3　SVM 算法与软件缺陷预测联系

软件缺陷预测模型本质上是一个模式识别过程,它的核心是分类问题,即模式识别问题。本文引入 SVM 的思想来实现软件缺陷预测。

一个软件系统是由软件模块依据一定的规则构成的,而模块是由各种复杂性度量属性(如总算子数目、McCabe 循环复杂度等)来描述其内在的结构特点的。因此,软件模块中所蕴涵的缺陷数与复杂性度量属性间存在着某种映射关系。利用 SVM 算法,对一部分有出错倾向的软件模块属性值进行训练,使其与软件模块的出错概率对应起来,形成一个黑箱模型。运用此模型来预测其余软件模块的出错倾向,即可达到预测软件缺陷的目的。

2　软件缺陷预测模型的建立

2.1　软件缺陷预测模型

基于 LASSO-SVM 方法建立软件缺陷预测模型的流程如图 3 所示。其具体步骤如下:

a)获取软件复杂性度量属性数据集;

b)消除数据集中存在的错误数据,运用 LASSO 方法来降维,获得精简的属性子集;

c)搭建 SVM 网络系统框架;

d)用交叉验证算法结合 LASSO 特征选择方法获得的属性子集训练 SVM 网络相关参数;

e)将测试样本输入训练好的 SVM 网络系统来进行软件缺陷预测;

f)实验结束,输出软件缺陷预测分类结果。

2.2　LARS 算法

关于 LASSO 方法的求解问题虽然可以采用向前回归解决,但是效果一直不令人满意,直到 Efron 等人[12] 于 2004 年提出 LARS 算法。LARS 算法是每次在回归残差的基础上选择新的变量,该过程是一个残差不断减小的过程,具有很高的计算效率。回归残差综合了类标签变量与已选变量的信息,将该算法引入 LASSO 问题中,能高效地求得最优解,即所要求的变量选择序列。

LARS 算法的基本思想是:首先选择一个与响应变量相关性最大的观察变量,然后沿这个方向走一定的长度,直到出现

第二个观察变量。这两个观察变量与残差的相关性相同,就沿着这两个变量等角度的方向继续走。以此类推,选择出需要的观察变量。

算法步骤描述如下:

input:数据集 D

output:与类标签关联性强的特征序列

将所有的回归系数 $\hat{\beta}_j$ 设置为0,然后找到与响应变量 y_i 最相关的一个观察变量,记为 x_{j1};

将 $\hat{\beta}_j$ 沿着此观察变量的方向最大限度地增大,直到找到另一个与当前残差 $r = y - \hat{\mu}_i$ 最相关的观察变量被选中为止,记为 x_{j2};

将 $\hat{\beta}_j$ 朝着与 (x_{j1}, x_{j2}) 等角度的方向增大,直到出现第三个观察变量 x_{j3} 符合与当前残差最相关的这一条件;

之后一直沿着与 (x_{j1}, x_{j2}, x_{j3}) 等角度的方向增大,直到出现第四个观察变量,以此类推。

将上述算法应用到软件复杂性度量属性集中,可以得到一个特征序列。随着约束值 t 的变化,可以从路径中移除一些不重要的属性,从而达到降低复杂性度量属性集维数的目的。

2.3 SVM 算法最佳参数的选择

SVM 算法中惩罚参数 c 和核函数参数 g 的优化,目前国际上还没有形成一个统一的模型[13],也就是说最优算法参数选择并没有理论的指导,还只能是凭借经验、实验对比、大范围地搜寻等方法解决。这些方法需要付出较高的时间代价,并且选择效果无法保证。

交叉验证(cross validation)算法是用来验证分类器性能的一种统计分析方法,基本思想是在某种意义下将原始数据(dataset)进行分组,一部分作为训练集(train set),另一部分作为验证集(validation set)。其方法是首先用训练集对分类器进行训练,再利用验证集来测试训练得到的模型,以得到的分类准确率作为评价分类器的性能指标。

本文利用 10 折交叉验证(10-fold cross validation)算法来进行支持向量机的惩罚参数 c 和核函数参数 g 的优化。将训练集均分成 10 组,将每个子集数据分别作一次验证集;同时其余的九组子集数据作为训练集对 SVM 分类器进行训练,取最后得到的所有分类准确率的平均数,设为 cv,则该算法的伪代码如下:

```
start
% 相应的数据初始化
   bestAccuracy = 0;
   bestc = 0;
   bestg = 0;
% 将 c 和 g 划分网格进行搜索
   for c = 2^(cmin):2^(cmax)
   for g = 2^(gmin):2^(gmax)
% 采用 10-CV 方法
将 train 均分为 10 组,
记 train(1),train(2),…,train(10)。
相应地标签也要分离出来,
记为 train_label(1),train_label(2),…,train_label(10)
      for run = 1:10
让 train(run)作为验证集,其他的作为训练集。
      记录此时的验证准确率为 acc(run)
      end
      cv = (acc(1) + acc(2) + …acc(k))/k;
      if(cv > bestAccuracy)
         bestAccuracy = cv;bestc = c;bestg = g;
         end
   end
end
over
```

其中:c_{min}、c_{max}、g_{min}、g_{max} 都是给定的数。由于必须将 c 和 g 的值进行离散化查找,这里将 c 和 g 在 2 的指数范围网格内进行查找。

3 基于 LASSO-SVM 的软件缺陷预测模型实验

3.1 实验环境及实验数据

实验所用算法利用 MATLAB 2008a 的相关工具箱搭建,程序在配置为 Pentium(R) 4 CPU 3.00 GHz 2.99 GHz,504 MB 内存的电脑上运行。

实验数据来源于 NASA 的 MDP(metric data program)软件度量项目中 PC1 数据集,可从网址 http://mdp.ivv.nasa.gov 中下载。PC1 数据集来自于一个飞行器软件系统,用 C 语言编写代码,包含了 1 108 个子程序。

3.2 实验数据预处理

NASA 的软件度量数据集已经被广泛地应用于研究中,但数据集本身可能存在的问题却很少有学者直接涉及研究。本文参考英国赫特福德大学计算机科学学院 Gray 等人[14]的最新研究成果,其在文献中指出了 NASA 的数据集中存在着大量的"重复数据"和"矛盾数据",经过研究这些数据已经严重影响了软件缺陷预测的实验结果。

本文首先对 PC1 数据集进行了删除"重复数据"和"矛盾数据"的操作,经过实验,最终将其子程序的个数删减为 953 个。

接着对样本数据运用最小—最大规范化方法进行了归一化处理,将数据样本的每个属性的值映射到[0,1]区间,消除了因量纲不同可能引发的影响。

最后对经过归一化操作的数据运用 LASSO 方法进行特征降维,消除影响结果小的属性,得出的最终实验仿真结果如图 4 所示。

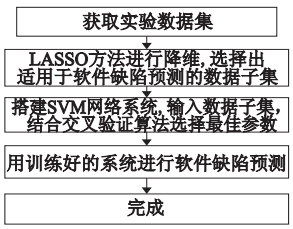

图3 建立软件缺陷预测模型的流程

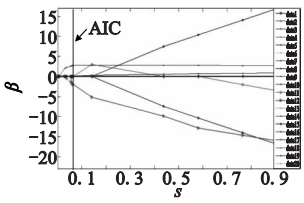

图4 LASSO实验结果

从图 4 中可以看出,最初有 21 个属性变量,随着 s 的变小(即约束条件 t 的变小),回归系数 β 逐步减少,最终变为全部为0。本文采用 AIC 准则作为选择回归系数的标准(参考文献)。AIC 准则即赤池信息准则,该准则运用下面的统计量评价模型的好坏:$AIC = -2L/n + 2K/n$。其中:L 是对数似然值;n 是观测值数目;k 是被估计的参数个数;AIC 准则要求其越小越好。按照 AIC 准则,本文最终选取了七个属性变量,即变量 5、6、9、10、17、20、21。

3.3 参数寻优

采用 MATLAB 2008a,参考 2.3 节所写的 10-CV 算法伪代码编程实现 SVM 算法中的惩罚参数 c 和核函数参数 g 的寻优

过程。具体实验时从经过 LASSO 方法选择后的数据集中随机选择 400 组子程序作为输入训练数据集,之后调用 MATLAB 的 SVMcgForClass 工具箱并编制相关的程序来实现两个参数的寻优。具体实验结果如图 5 所示。

从图 5 中可以看出,选择出的 SVM 最优惩罚参数 $c = 5.6569$,核函数参数 $g = 2$,分类准确率 = 93.25%。

3.4　实验运行

本文采用准确度、查准率、查全率和 $F1$ 值[15]来评价软件缺陷预测模型的预测能力。这些度量来自表 1。

表 1　交叉矩阵

模块	实际值	预测值
容易出错模块	正确的正例(TP)	错误的负例(FN)
不易出错模块	错误的正例(FP)	正确的负例(TN)

表 1 中:实际正例个数,$P = TP + FN$;实际负例个数,$N = FP + TN$;实例总数,$C = P + N$。

1)准确度(accuracy)　预测结果与实际结果相符合的模块个数占整个测试集的比例。

$$\text{accuracy} = \frac{TP + TN}{C}$$

2)查准率(precision)　预测为易错的模块中实际为易错模块所占比例。

$$\text{precision} = \frac{TP}{TP + FP}$$

3)查全率(recall)　易错模块被正确识别的比例。

$$\text{recall} = \frac{TP}{P}$$

4)F-度量(F-measure)　precision 和 recall 的调和平均数。

$$F1 = \frac{1}{\frac{2}{\text{presion}} + \frac{1}{\text{recall}}}$$

采用剩下的 553 组子程序作为测试集来进行实验,仿真实验运行结果如图 6 所示。

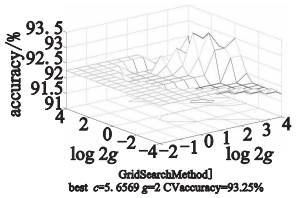

图5　SVM参数选择结果

图6　LASSO-SVM软件缺陷预测结果

从实验运行结果可以看出:$TP = 32$,$FN = 9$,$FP = 16$,$TN = 496$。

结合表 1 所示的交叉矩阵得到:准确度 = 93.25%,查准率 = 66.67%,查全率 = 78.04%,F-度量 = 72.72%。

3.5　实验结果与分析

将基于本文方法(LASSO-SVM)与 Fisher 线性判别分析(LDA)[16]、聚类分析(CA)[17]、BP 神经网络(BPNN)[18]和逻辑回归(LR)[19]等方法建立的软件缺陷预测模型进行比较,结果如表 2 所示。

表 2　本文方法与传统方法对比

预测模型	准确度/%	查准率/%	查全率/%	$F1$ 值/%
LDA	55.1	56.4	55.7	56.5
CA	72.7	64.6	72.3	68.2
BPNN	80.4	61.7	73.5	67.1
LR	85.3	62.6	70.7	66.4
LASSO-SVM	93.25	66.7	78.0	72.7

从表 2 中可以看出,基于 LDA 方法建立的软件缺陷预测模型效果很差,因为该方法是基于线性分类函数进行缺陷预测,而软件复杂性度量属性值基本都是非线性的;基于 CA 和 BPNN 方法的预测性能明显地好于 LDA 方法,但这两种方法主要依靠人为经验,泛化能力较差;基于 LR 方法的预测性能比较好,但该方法受到样本容量的限制,只有样本容量大时能有很好地预测效果,且预测时间较长。

本文方法在各项指标上均优于传统的软件缺陷预测算法,这是因为本文利用 LASSO 特征选择方法通过残差拟合将影响不显著变量的回归系数直接置为 0,减少了用于实验的复杂性度量属性集;之后利用 10 折交叉验证方法结合 LASSO 筛选出的属性子集对 SVM 算法的相关参数进行寻优,加速了 SVM 网络收敛速度和效率,提高了网络分类精度,很好地实现了软件缺陷预测。

本文所提方法还存在着以下几点不足:

a)本文采用 LARS 算法来解决 LASSO 方法所涉及的问题,LARS 算法适用于低维数据空间。倘若遇到软件复杂性度量属性集远远大于测试历史数据的情况,即 $m >> n$,在这种情况下,需要寻找其余的算法来处理 LASSO 问题。

b)本文所提方法解决的是二分类情况下的软件缺陷预测问题。倘若软件复杂性度量数据集采用其余方式,如缺陷的严重程度等来描述缺陷问题,则本文方法不适用。

c)交叉验证选择 SVM 最优参数时有一个问题存在,就是可能会有多组参数对应最高的验证分类准确率,这种情况的处理措施是下一步的研究方向。

d)交叉验证寻优时需要多次训练样本,计算成本仍然需要进一步减少。

4　结束语

本文的贡献是将一种新的特征选择方法引入到软件缺陷预测中。针对原始软件复杂性度量数据集中存在着大量对分析结果影响甚微的复杂性度量属性集,本文用 LASSO 方法将大多数属性集的回归系数置为零,仅保留了一小部分对分析结果影响巨大的属性子集。同时本文用 SVM 算法对处理后的数据集进行软件缺陷预测,并通过与其他软件缺陷预测方法相对比,验证了该方法的有效性和优越性。实验结果表明,该方法比传统软件缺陷预测方法具有更好的预测速度和预测效果。下一步的研究方向是拓展本文所提方法的适用范围,并且进一步研究 SVM 最优参数的选择问题。

参考文献:

[1]　CHALLAGULLA V U B, BASTANI F B, YEN I L, *et al*. Empirical assessment of machine learning based software defect prediction techniques[C]//Proc of the 10th IEEE International Workshop on Object-Oriented Real-time Dependable Systems. Washington DC: IEEE Computer Society, 2005: 263-270.

[2]　王琪. 软件质量预测模型中的若干关键问题研究[D]. 上海:上海交通大学,2006.

[3]　崔正斌. 软件可靠性预测技术研究[D]. 郑州:信息工程大学,2010.

[4]　毛勇,周晓波,夏铮,等. 特征选择算法研究综述[J]. 模式识别与人工智能,2007,20(2):211-218.

[5]　蒋胜利. 高维数据的特征选择与特征提取研究[D]. 西安:西安电子科技大学,2011.

(下转第 2754 页)

于解释规约,希望系统包括的细节越少越好,通过画出多种不同的谓词行为图来解释系统的规约。

2.2 形式化描述

首先精确定义谓词行为图的 TLA 公式。形式上,一个谓词行为图包含一个有向图、一个从初始节点开始的节点子集。这里每一个节点都以谓词来标记,而每个边都以动作标记。假定一个给定状态函数 v 的谓词行为图并作以下解释:N 为节点集;I 为初始节点集;$E(n)$ 为从节点 n 出发的边集;$d(e)$ 为边 e 的目的节点;P_n 为标记节点 n 的谓词;ε_e 为标记边 e 的行动。

通过以下的定义得到公式△:

$$\mathrm{Init}_{\triangle} \triangleq \exists n \in I : P_n$$

$$A_n \triangleq \exists e \in E(n) : \varepsilon_e \wedge P'_{d(e)}$$

$$\triangle \triangleq \mathrm{Init}_{\triangle} \wedge \forall n \in N : \Box[P_n \Rightarrow A_n]_v$$

当没有确定的标签附属边 e,则认为 ε_e 为真。当没有明确指定初始节点集,本文把 I 当做是 N,依据常用空集的习惯,A_n 当且仅当没有从节点 n 出发的边。

2.3 谓词行为图的证明

通常所说规约 Π 的谓词行为图表明 Π 蕴涵代表谓词行为图的 TLA 公式△,公式 Π 一般具有 $\mathrm{Init}_{\Pi} \wedge \Box[M]_u \wedge L$ 的形式,这里的 L 是公平性条件,公式 $\triangle \equiv \mathrm{Init}_{\triangle} \wedge \forall n \in N : \Box[P_n \Rightarrow A_n]_v$。为证明 $\Pi \Rightarrow \triangle$,需证明:

a) $\mathrm{Init}_{\Pi} \Rightarrow \mathrm{Init}_{\triangle}$;

b) 对每个节点 n,$\mathrm{Init}_{\Pi} \wedge \Box[P_n \Rightarrow A_n]_v$。

条件 a)是对谓词的断言,这个很容易证明。为证明条件 b),通常要找到一个不变量 Inv,使得 $\mathrm{Init}_{\Pi} \wedge \Box[M]_u \Rightarrow \Box \mathrm{Inv}$,因此 $\Pi \Rightarrow \Box[M \wedge \mathrm{Inv}]_u$,通过 $[M \wedge \mathrm{Inv}]_u \Rightarrow [P_n \Rightarrow A_n]_v$ 对每个节点 n,通常 u 和 v 是元组,且每个 v 的组件是 u 的组件。在这种情况下,则需以下公式成立:对每个节点 n,定义动作 A_n,有 $M \wedge \mathrm{Inv} \Rightarrow [P_n \Rightarrow A_n]_v$,从而证明对每个节点 n:

$$P_n \wedge M \wedge Inv \Rightarrow (\exists m \in E(n) : \varepsilon_m \wedge P'_{d(m)}) \vee (v' = v)$$

断言 P_n 且 Inv 为真是一个 M 步,同时改变 v 是一个 ε_m,最后对从节点 n 出发的某些边,以满足 $P_{d(m)}$ 的状态结束。

3 结束语

本文主要针对使用 TLA 公式规约复杂转移系统时存在复杂难以理解的不足,提出使用谓词行为图对转移系统进行描述,并用 TLA 规约进行解释与证明,说明了 TLA 对系统的规约与使用谓词行为图描述系统具有等价性。尽管谓词行为图具有直观易懂的优点,但在处理复杂并发转移系统时存在图形复杂。因此,需要将复杂系统分解成简单模块,使用谓词行为图分别对模块进行描述,合并讨论整个系统的正确性,这是下一步的研究工作。

参考文献:

[1] LAMPORT L. The temporal logic of actions[J]. ACM Transactions Programming Languages and Systems,1994,16(5):872-923.

[2] LAMPORT L, MALKHI D, ZHOU Li-dong. Reconfiguring a state machine[J]. SIGACT News,2010,41(1):63-73.

[3] LAMPORT L. Specifying systems [M]. Boston: Addison-Wesley, 2003.

[4] 唐郑熠,李均涛,李祥. 行为时序逻辑中公平性的研究与完善[J]. 计算机应用研究,2010,27(5):1788-1790.

[5] 夏薇,姚益平,慕晓冬. 基于 TLA 的事件图模型形式化验证方法[J]. 计算机应用研究,2011,28(11):4171-4173,4187.

[6] KAYNAR D,LYNCH N,SEGALA R, *et al*. The theory of timed I/O automata[M]. [S. l.]:Morgan & Claypool Publishers,2005.

[7] CHENG Hong, LO D, ZHOU Yang, *et al*. Identifying bug signatures using discriminative graph mining[C]//Proc of the 18th International Symposium on Software Testing and Analysis. 2009:141-152.

[8] MARCZAK W R, ALVARO P, CONWAY N, *et al*. Confluence analysis for distributed programs: a model-theoretic approach technical report, No UCB/EECS-2011-154[R]. UC Berkeley: EECS, 2011.

[9] GUREVICH Y. Sequential abstract state machines capture sequential algorithms[J]. ACM Trans on Computational Logic,2000,1(1): 77-111.

[10] 黄贻望,万良,李祥. 基于 TLA 的 NS 安全协议分析及检测[J]. 计算机工程与科学,2010,32(7):38-41.

(上接第 2751 页)

[6] 梁斌,陈敏,缪柏其,等. 基于 LARS-LASSO 的指数跟踪及其在股指期货套利策略中的应用[J]. 数理统计与管理,2011,30(6): 1104-1114.

[7] HALSTEAD M H. Elements of software science[M]. New York: Elsevier North Holland,1977.

[8] McCABE T J. A complexity measure[J]. IEEE Trans on Software Engineering,1976,SE-2(4):308-320.

[9] 李轩,郝克刚,葛玮. 面向对象软件度量的分析和研究[J]. 计算机技术与发展,2006,16(11):38-41.

[10] HASTIE T, TAYLOR J, TIBSHIRANI R, *et al*. Forward stagewise regression and the monotone lasso[J]. Electronic Journal of Statistic,2007,16(1):1-29.

[11] 何俊学. 基于支持向量机的软件可靠性模型研究[D]. 兰州:兰州理工大学,2009.

[12] EFRON B,HASTIE T,JOHNSTONE I,*et al*. Least angle regression [J]. Journal of the Institute of Mathematical Statistics,2004,32 (2):407-499.

[13] 姜慧研,宗茂,刘相莹. 基于 AC0-SVM 的软件缺陷预测模型的研究[J]. 计算机学报,2011,34(6):1148-1154.

[14] GRAY D, BOWES D, DAVEY N,*et al*. The misuse of the NASA metrics data program data sets for automated software defect prediction [C]//Proc of the 15th Annual Conference on Evaluation & Assessment in Software Engineering. 2011:96-103.

[15] WANG Xu-hui,SHU Ping,CAO Li,*et al*. A ROC curve method for performance evaluation of support vector machine with optimization strategy[C]//Proc of International Forum on Computer Science. Technology and Applications. 2009:117-120.

[16] 姚全珠,宋志理,彭程. 基于 LDA 模型的文本分类研究[J]. 计算机工程与应用,2011,47(13):150-154.

[17] 衷湾,余都,江顺亮. 基于 CA 的离散粒子系统仿真和显示研究[J]. 计算机与现代化,2011(1):1-5,10.

[18] 林盾,陈俐. BP 神经网络在模拟非线性系统输出中的应用[J]. 武汉理工大学学报,2003,27(5):731-734.

[19] BRIAND L C,MELO W L,WUST J. Accessing the applicability of fault-proneness models across object-oriented software projects[J]. IEEE Trans on Software Engineering,2002,28(7):706-720.