基于贝叶斯网络预测克隆代码质量

刘冬瑞 刘东升 张丽萍 侯 敏 王春晖
(内蒙古师范大学计算机与信息工程学院 呼和浩特 010022)

摘 要 针对软件中克隆代码的质量进行研究,评价软件当前所有版本中克隆代码的质量。在此基础上使用贝叶斯网络训练已有样本数据,得到克隆代码质量预测模型,其能预测软件未发布版本中克隆代码的质量,根据评价和预测结果给开发人员提供克隆代码重构和有效复用的建议,防止有害克隆代码的大量繁殖。实验表明,该方法能够较准确地预测软件中克隆代码的质量。

关键词 克隆代码,贝叶斯网络,质量评估模型,预测,重构
中图法分类号 TP311.5 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.04.036

Prediction of Code Clone Quality Based on Bayesian Network

LIU Dong-rui LIU Dong-sheng ZHANG Li-ping HOU Min WANG Chun-hui
(College of Computer and Information Engineering, Inner Mongolia Normal University, Hohhot 010022, China)

Abstract This paper researched on the quality of code clone in the softwareand , evaluated the code clone quality of the current versions. Then on this basis, Bayesian network was used to train the existing sample data gets the prediction model of code clone which is able to predict the quality. The prediction results are able to help developers make judgments that code clone should be reconstructed or efficiently reused. The experiment shows that the method can be used to predict the quality of code clone in software more accurately.

Keywords Code clone, Bayesian network, Quality evaluation model, Prediction, Reconstruct

1 引言

随着软件体系结构、设计模式和相关领域研究的不断深入,人们在开发新的系统中不必总重复创造别人在过去已经创造过的东西,而可以在软件开发的各个层次上复用已有的成果[1]。软件复用过程中产生的大量代码被称作克隆代码[2],简称为克隆,克隆给软件开发与维护带来的影响具有双重性。复制一段高质量的代码不仅可以降低编写新代码的潜在风险,而且可以提高开发效率,是有益克隆[3];但是复制一段低质量的代码就有可能引入新的 Bugs[3]。

对于软件中的克隆代码,早期研究的重点在于找出克隆并通过重构将其消除[4]。近年来,多项研究发现有些克隆代码有利于软件的开发与维护[5],不一定要对其重构并消除,而可以将这些克隆封装为适合软件系统使用的可复用构建,但构建必须保证质量[6]。因此亟需一种评估克隆代码质量的方法,在软件复用过程中实现对克隆代码的有效管理,为软件复用提供有效的保障。

2 相关技术描述

克隆管理[7]就是采用多种技术,有效利用一些高质量的克隆而尽量避免低质量的克隆,充分利用克隆代码的积极作用而降低它的负面效益。克隆管理过程涉及克隆重构、克隆代码质量评估、机器学习等技术[7]。

2.1 克隆重构

克隆代码质量评估与克隆重构有着紧密的联系[8]。Manishankar Mondal 等人提出一种基于挖掘辅助规则的克隆重构自动排序方法[9],该方法能够确定克隆片段被重构的重要性,并且设计了一个特殊的模型 SPCP(Similarity Preserving Change Pattern),认为因该模型而发生变化的克隆片段就是需要重构的克隆片段"候选人",这些克隆片段需要重点考虑。

Fowler 描述了 72 种去除代码味道的重构模型[10]。随着时间的推移,重构模型的数量已经增长到 93 种,现在已经存在一个描述了所有重构模式的重构目录[11]。早期的研究发现[12]:在所有的重构模式中,去除方法、牵引方法、抽取特殊

到稿日期:2015-11-30 返修日期:2016-02-27 本文受国家自然科学基金(61363017,61462071),内蒙古自然科学基金资助项目(2014MS0613),内蒙古自治区高等学校科学研究项目(NJZY14039),内蒙古师范大学科学研究项目(2013ZRYB06)资助。

刘冬瑞(1989—),男,硕士生,主要研究方向为软件分析,E-mail:15326003430@163.com;刘东升(1956—),男,教授,CCF 会员,主要研究方向为软件工程、计算机教育;张丽萍(1974—),女,教授,CCF 会员,主要研究方向为软件工程、软件分析;侯 敏(1979—),女,讲师,主要研究方向为软件分析、计算机教育;王春晖(1979—),女,讲师,CCF 会员,主要研究方向为软件分析、多媒体、计算机辅助教学。

类和名称重构等模式很适合应用在克隆重构方面，这些重构模型的具体细节在重构目录和其他研究中都可以找到[13]。

2.2 克隆代码质量评估

Akito Monden 等人对克隆代码和软件质量之间的关系进行了研究[14]。结果发现大多数情况下块克隆比非块克隆更加稳定，只有当代码量超过 200SLOC 时非块克隆才比块克隆稳定，因此一般情况下包含块克隆的软件代码质量较高，不建议对其重构。Radhika D. Venkatasubramanyam 等人使用 EMISQ 质量评估模型[15]评估克隆代码的质量，给出了克隆代码重构的优先排序建议[16]。从维护费用(MO)、软件质量缺陷(LSQ)和重构费用(RFT)3 个方面提供克隆代码质量评估参数，最终完成软件代码重构等级的排序，给维护人员提供了一个重要的重构建议。但是该研究只针对已经发布的软件进行克隆代码质量分析，并没有阻止有害克隆代码的繁殖。

2.3 贝叶斯网络

贝叶斯网络是一个有向无环图，可以表示变量集合的联合概率分布，能够分析变量之间的相互关系，并利用贝叶斯定律和统计推断方法来实现预测和分类的任务。图 1 所示的贝叶斯网络[17]中，play 节点为根节点并且是 outlook 节点、windy 节点、humidity 节点和 temperature 节点的父节点，可将从节点 play 指向节点 windy 的有向边视为 play 被 windy 影响，即是否出去玩与是否有风有着概率的关系。

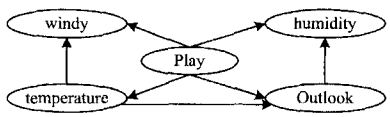

图 1 贝叶斯网络

本文首次将贝叶斯网络方法应用到克隆代码质量评估领域中，在开发软件新版本的过程中，为开发人员提供代码复用和重构的建议，可以预防低质量克隆代码的繁殖，同时提高高质量克隆代码的复用效率，在克隆管理领域中实现了预防克隆管理[7]的任务。

3 克隆代码质量预测

预测克隆代码质量研究的具体步骤分为：1)提取评估度量；2)训练克隆代码质量预测模型；3)预测克隆代码质量；4)评估预测模型。

3.1 提供评估度量

本研究采用 EMISQ 软件内部质量评价模型评价克隆代码的质量，该模型是以 ISO14598 质量模型为基础的软件内部质量评测方法。国际上已经有很多项目的应用证实了该方法的实用价值[16]。从维护费用(MO)和软件质量缺陷(LSQ)两方面评价克隆代码质量。由于克隆代码重构的具体操作可能无法实现自动化，往往需要开发人员的涉入，因此不考虑重构费用(RFT)。

3.1.1 维护费用(MO)

依靠克隆检测[3]结果和克隆映射[18-20]关系中获取的相关度量值来计算克隆代码的维护成本。具体度量值描述如下。

(1)克隆代码的行数(MM^{NL})：克隆群[3]中克隆代码行数越多，说明该克隆群的维护开销越大。

(2)克隆群改变频率(MM^{FCF})：克隆群的改变频率越高，说明其中的克隆代码发生不一致改变[19]的可能性越大。

(3)克隆群上次改变的时间(MM^{DLFC})：如果某克隆群在近期发生过改变，那么该克隆群再次发生改变的可能性较大。

(4)克隆寿命(MM^{CA})：一个克隆群存在的时间越长，说明其越稳定。

利用上述属性，在质量模型(EMISQ)的基础上评价克隆代码的可维护性指数(MM^{MQM})。克隆片段的维护费用的度量值计算过程如下：首先把分布在不同版本中的同一克隆群的维护费用分别进行归一化处理。归一化克隆群的维护费用值($MM^{e}_{C(n,j)}$)可以用式(1)表示。

$$MM^{e}_{C(n,j)}=\frac{|MM^{e}_{C(n,j)}|}{\mathrm{Max}(MM_a)} \tag{1}$$

其中，$C(n,j)$指在第 n 个版本中的第 j 个克隆群。然后对上面得到的同一克隆群的若干归一化费用值($MM^{e}_{C(n,j)}$)取和，即可得到该克隆群维护费用的值，可以用式(2)表示。

$$MO(C(n,j))=\sum(MM^{e}_{C(n,j)}) \tag{2}$$

3.1.2 克隆代码质量缺陷(LSQ)

通过不同的静态分析规则来确定克隆代码发生 bug 的可能性。本研究用静态分析工具 FlawFinder[21]提取如下静态分析规则的值。

(1)规则的严格等级($Level_i$)[22]：该值越低，引起 bug 的可能性也就越低。

(2)规则的权重(W_i)：违反静态规则的权重。

(3)规则的临界($Crit_i$)：将违反静态分析规则的等级($Leveli$)和权重(W_i)结合起来分析。

(4)克隆群违反规则的数量($CoV^{i}_{(n,j)}$)。

克隆群的 LSQ 可以用式(3)进行计算：

$$LSQ(C_{(n,j)})=\frac{\sum_{i=1}^{I}Crit_i * CoV^{i}_{(n,j)}}{\sum_{i=1}^{I}CoV^{i}_{(n,j)}} \tag{3}$$

其中，$\sum_{i=1}^{I}CoV^{i}_{(n,j)}$代表第 n 个版本中第 j 个克隆群违反规则的数量，$\sum_{i=1}^{I}Crit_i * CoV^{i}_{(n,j)}$是克隆片段违反静态规则数量的加权和。

3.2 训练克隆代码质量预测模型

本文使用 Weka[17]训练克隆代码质量预测模型。预测模型中每一个质量评估度量对应一个贝叶斯网络节点，其中克隆寿命 CA 的计算方法描述如下。

算法 1 克隆寿命计算方法

对于第 n 个版本 V_n 的每一个克隆群 $CG_{n,i}$

对 $CG_{n,i}$提取源码 C_i

对于第 n+1 个版本 V_{n+1}有映射关系的克隆群 $CG_{n+1,j}$

对 $CG_{n+1,j}$提取源码 C_j

比较 C_i 与 C_j 的相似度，将相似度存储在 Sim 中：

IF Sim＞阈值，该克隆群寿命 CA 增加 1 个单位，即 CA＝CA＋1；

ELSE CA＝1

返回 CA

贝叶斯网络节点使用的是离散变量，因此把所有评估度量值离散化为 VeryHigh，High，Mid 和 Low 4 种状态。

3.3 预测克隆群质量

预测软件未发布版本中克隆群的质量情况具体分为两种：1)在不知道任何节点信息的情况下，预测贝叶斯网络中某个节点发生的概率；2)在已知某些度量节点值的情况下，可以预测网络中某个结果节点发生的概率。

3.4 评估预测模型

国内外已经有很多评价贝叶斯网络性能的研究[23]，其中最著名的是 Cooper 和 Herskovits 提出的 K2 算法[24]，其应用贝叶斯评分[23]和爬山算法进行搜索的方法来优化网络模型，其中的评分函数即为贝叶斯评分的一种简化，又称为 K2 评分。本文采用 K2 评分作为评价贝叶斯网络的测度，并且结合分类器评价贝叶斯网络节点分类的性能。使用分类器中如下的输出值作为评价指标。

(1)CCI：正确分类实例[17]；

(2) Kappa 值[17]：评判分类器的分类结果与随机分类的差异度，$K=1$ 表明分类器完全与随机分类相异，所以该值越接近 1 则分类效果越好；

(3)案例覆盖度(Coverage)[17]：节点覆盖率；

(4)TP：显示真阳性率，即正确的正例；

(5)FP：显示假阳性率，即错误的正例；

(6)FN：错误的反例；

(7)Precision：查准率，$Precision=TP/(TP+FP)$；

(8)Recall：查全率，$Recall=TP/(TP+FN)$；

(9) F 值：$F=2*Recall*Precision/(Recall+Precision)$；

(10)CLASS：显示类别标签。

4 实验结果与分析

本研究选用 3 款 C 语言软件，共计 71 个稳定版本作为实验数据，如表 1 所列。

表 1　目标软件

软件	Xorriso	Smalltalk	Bison
平均大小/MB	8.08	20.1	14.12
版本数目	29	18	24
(平均)克隆群数	61	61	31

使用贝叶斯网络对软件中所有离散化度量进行训练，得到克隆群代码质量预测模型，图 2 示出了通过 Xorriso 软件训练的预测模型。在图 2 中，节点 LSQ 值为 High 的概率是 0.25，为 Mid 的概率是 0.38，为 Low 的概率是 0.36，说明这 29 个版本的 xorriso 软件中有近 26%的克隆群的质量较差。Refac 节点值为 Y 的概率是 0.14，为 N 的概率是 0.85，该软件中接近 15%的克隆群需要重构。

在任意给定 NL，DLFC，MQM，FCF，CA，Cov，L1，L2，L3，L4，L5 值的情况下，LSQ，MO 和 Refac 值的概率即预测会受到影响。当某个克隆群具备特征 $D=\{$High，Mid，Mid，Low，High，VeryHigh，Mid，Low，Low，High，High$\}$ 时，实验预测出 LSQ 值为 High 的概率是0.92，MO 值为 Low 的概率是 0.58，Refac 值为 Y 的概率是 0.9975，说明该克隆群具有高的质量缺陷等级，即该克隆群的质量较差，并且维护该克隆群的费用较低，因此预测出该克隆群需要被重构。

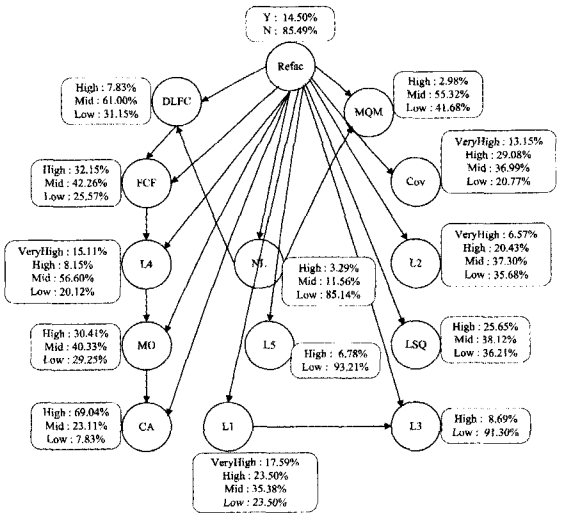

图 2　克隆群质量预测模型

使用 K2 算法与分类器评价贝叶斯网络性能的指标如表 2 所列，Refac 节点评价指标如表 3 所列。

表 2　贝叶斯网络评价指标

软件	CCI/%	Kappa	Coverage/%
Xorriso	97.28	0.8877	99.6
Smalltalk	99.78	0.9943	100
Bison	98.86	0.9704	100

表 3　Refac 评价指标

软件	TP	FP	Precsion	Recall	F	CLASS
Xorriso	0.882	0.012	0.926	0.882	0.904	Y
	0.988	0.118	0.980	0.988	0.984	N
Smalltalk	0.992	0.000	1.000	0.992	0.996	Y
	1.000	0.008	0.997	1.000	0.999	N
Bison	0.995	0.014	0.962	0.995	0.978	Y
	0.986	0.005	0.998	0.986	0.992	N

4.1 实验结果分析

从表 2 中可以看出，克隆质量预测模型的 Kappa 值接近于 1，表明分类器几乎完全与随机分类相异，CCI 值大于 97.28%，表明克隆群质量预测模型中的 14 个度量节点基本都是正确分类的；Coverage 值大于 99.6%，说明该分类器在分类过程中基本覆盖全部 14 个度量节点。

从表 3 中可以看出，Refac 值为 Y 的查准率、查全率和 F 值在 0.88～1 之间；Refac 值为 N 的查准率、查全率和 F 值大于 0.98，说明该质量预测模型能够准确、有效地预测克隆代码质量。

4.2 研究的不足

本研究主要存在以下不足：

(1)在评价克隆代码质量的过程中借鉴了质量评价工具，因此提取评价度量时需要研究人员的涉入，不能完全自动地评价克隆代码质量。

(2)贝叶斯网络的训练参数是离散变量，训练质量预测模型以前需要根据经验为每个节点变量设定阈值范围使其离散

化，阈值范围的选取直接影响着贝叶斯网络的训练结果。

结束语 本文针对开源的 C 语言软件进行实验，有效预测了克隆代码质量，并且给出了克隆代码的重构建议。在开发新版本软件的过程中预测出克隆代码的质量，推荐质量差的克隆代码进行重构操作，能够防止有害克隆的繁殖，同时推荐质量高的克隆代码封装为可复用构建，从而提高了软件开发的效率。

本文提出的方法同样适用于其他常用的编程语言。接下来将针对 Java，Python，C＃等常用语言的软件进行研究，并且把克隆代码质量预测模型封装到 Web 项目中，将该研究应用到工业领域进行测试和完善。

参 考 文 献

[1] ZHANG S K, WANG L F, YANG F Q. Architecture Based Software Developing Model[J]. World Sci-Tech, 1999, 21(3): 31-35(in Chinese)
张世琨，王立福，杨芙清. 基于体系结构的软件开发模式[J]. 世界科技研究与发展，1999，21(3)：31-35.

[2] ZHANG R X, ZHANG L P, WANG H, et al. A Novel Approach for Clone Group Mapping by using Topic Modeling [J]. International Journal of Software Engineering & Applications, 2015, 6(2).

[3] ROY C K. Detection and analysis of near-miss software clones [C]// IEEE International Conference on Software Maintenance. 2009: 447-450.

[4] BALAZINSKA M, MERLO E, Dagenais M, et al. Advanced Clone-Analysis to Support Object-Oriented System Refactoring [C]// Conference on Reverse Engineering. IEEE Computer Society, 2000.

[5] KAPSER C J, GODFREY M W. "Cloning considered harmful" considered harmful: patterns of cloning in software[J]. Empirical Software Engineering, 2008, 13(6): 645-692.

[6] WEI W U, ZHANG M X. Research on the Application of Software Reuse Technology Based on Components[J]. Journal of Shanxi Datong University (Natural Science Edition), 2009, 25(1): 8-10. (in Chinese)
武伟，张明新. 基于构件的软件复用技术应用研究[J]. 山西大同学报(自然科学版)，2009，25(1)：8-10.

[7] ZIBRAN M F, ROY C K. The Road to Software Clone Management: A Survey: Technical Report 2012-03[R]. The University of Saskatchewan, Canada, 2012: 1-66.

[8] CHATTERJI D, CARVER J C, KRAFT N A. Cloning: The need to understand developer intent[C]// 2013 7th International Workshop on Software Clones (IWSC). IEEE, 2013: 14-15.

[9] MANDAL M, ROY C K, SCHNEIDER K A. Automatic ranking of clones for refactoring through mining association rules[C]// IEEE Conference on Software Maintenance, Reengineering and Reverse Engineering. IEEE, 2014: 114-123.

[10] FOWLER M, BECK K, BRANT J, et al. Refactoring: Improving the Design of Existing Code[M]. Addison Wesley, 1999.

[11] ZIBRAN M F, ROY C K. Conflict-aware optimal scheduling of prioritized code clone refactoring[J]. IET Software, 2013, 7(7): 167-186.

[12] ROY K C, CORDY J. Near-miss function clones in open source software: an empirical study[J]. Journal of Software Maintenance & Evolution Research & Practice, 2010, 22(3): 165-189.

[13] LEE S, BAE G, CHAE H, et al. Automated scheduling for clone-based refactoring using a competent ga[J]. Softw. Pract. Exper., 2010, 41(5): 521-550.

[14] MONDEN A, NAKAE D, KAMIYA T, et al. Software Quality Analysis by Code Clones in Industrial Legacy Software[C]// IEEE International Symposium on Software Metrics. IEEE Computer Society, 2002: 1-8.

[15] PLÖSCH R, HENTSCHEL A, SAFT M, et al. The EMISQ method and its tool support-expert-based evaluation of internal software quality[J]. Innovations in Systems and Software Engineering, 2008, 4(1): 3-15.

[16] VENKATASUBRAMANYAM R D, Gupta S, SINGH H K. Prioritizing code clone detection results for clone management [C]// 2013 7th International Workshop on Software Clones (IWSC). IEEE, 2013: 30-36.

[17] FRANK E, HALL M, HOLMES G, et al. Weka-A Machine Learning Workbench for Data Mining[M]// Data Mining and Knowledge Discovery Handbook. Springer US, 2009: 1269-1277.

[18] KIM M, SAZAWAL V, NOTKIN D, et al. An empirical study of code clone genealogies[J]. ACM Sigsoft Software Engineering Notes, 2005, 30(5): 187-196.

[19] TU Y, ZHANG L P, WANG C H, et al. Clone genealogies extraction based on software evolution over multiple versions[J]. Journal of Computer Applications, 2015, 35(4): 1169-1173. (in Chinese)
涂颖，张丽萍，王春晖，等. 基于软件多版本演化提取克隆谱系[J]. 计算机应用，2015，3(4)：1169-1173.

[20] YUN L L, ZHANG L P, YU C H, et al. Predicting inconsistent change probability of code clone based on latent Dirichlet allocation model[J]. Journal of Computer Applications, 2014, 34(6): 1788-1791. (in Chinese)
尹丽丽，张丽萍，王春晖，等. 基于潜在狄利克雷分配模型预测克隆代码不一致变化的可能性[J]. 计算机应用，2014，34(6)：1788-1791.

[21] FlawFinder[OL]. http://www.dwheeler.com/flawfinder/.

[22] DALE M. Static analysis of the VoteHere VHTi reference implementation source code using Flawfinder and RATS[OL]. http://cisa.umbc.edu/courses/cmsc/444/fall05/studentprojects/dale.pdf.

[23] WEI Z Q, et al. Improved Bayesian Network Structure Learning with Node Ordering via K2 Algorithm[M]// Intelligent Computing Methodologies. Springer International Publishing, 2014: 44-55.

[24] COOPER G F, HERSKOVITS E. A Bayesian method for the induction of probabilistic networks from data[J]. Machine Learning, 1992, 9(4): 309-347.