

基于复杂网络的软件测度研究

马杰良¹ 郑伟才² 王俊超²

摘要

随着软件系统规模的增大,软件各组成元素之间的交互日益复杂,导致软件开发经常处于不稳定状态,软件质量问题逐步突出.借鉴复杂网络的发展成果,将软件结构描述为加权有向图,使用节点重要度定义了出度系数,以此来查找和判断软件结构中缺陷概率比较高的类,并利用无标度网络的缺陷发生特性来研究软件结构的缺陷分布.实例结果表明:该度量方法对查找软件缺陷类是有效的,有利于在软件开发初期就发现缺陷,提高软件开发质量.

关键词

无标度网络;节点重要度;节点团;软件模型

中图分类号 TP311.5

文献标志码 A

0 引言

Introduction

当前软件系统的规模越来越庞大,结构越来越复杂,而越复杂的软件就越难以保证其质量及生产率,使得软件开发经常处于失控状态.因此,如何在有限的时间内,提高软件开发的效率并保证软件开发质量,是当前亟待解决的问题之一.

到目前为止,对软件质量进行研究主要有:Mechelle等^[1]通过大量的实验研究发现软件有80%的功能集中在20%的代码中;Malaiya等^[2]则通过研究软件模块的大小和其缺陷密度的关系来刻画软件的可靠性.20世纪80年代末出现一些适合面向对象类软件的度量方法,较具代表性的是Chidamber等^[3]提出的C & K度量方法和后来Abreu^[4]提出的MOOD方法.但以上这些度量方法主要聚焦于内部复杂性,侧重微观上的统计,很难准确地刻画现代软件日趋复杂的结构特性.再则,当前复杂网络的研究成果为探索复杂软件系统的结构和行为特性提供了有力支持,软件网络观得到了越来越多研究者的认同.

有鉴于此,本文根据目前新兴的复杂网络研究成果,结合软件系统的内部拓扑结构特点,提出了一种从整体上对软件模型的类(模块)重要度进行评估的算法.

1 软件测度分析

Analysis of software measurement

随着软件的网络化趋势越来越明显,软件与网络的关系更加密不可分,已有研究人员采用实验仿真验证得出,软件系统是一个具有幂律分布的无标度网络^[5].这说明软件中功能和开发代价分布的不均衡性或异质性,也就是大部分的类只实现系统少部分的功能,而存在少部分与众多类有调用关系的高标度类,实现了系统的主要功能,一旦这些类产生问题,就会导致软件不可用,且构造这些类需要耗费整个项目80%的成本.

在基于复杂网络的研究方法中,有很多概念和方法可用来反映网络的相关统计特性,其中最重要的有节点的度分布^[6].在软件网络中,一个节点的度可以引申为该类在软件中被其他类调用的次数.因此,从直观上看,如果一个类被调用的次数越多,那么它的重要性就

收稿日期 2009-07-20

作者简介

马杰良,男,博士,副教授,硕士生导师,主要从事复杂网络,离散系统与图论方面的研究. njkjmjl@163.com

郑伟才(通讯作者),男,硕士生,主要研究方向为复杂网络,程序分析和软件工程. icefeng163@163.com

1 南京信息工程大学 电子与信息工程学院, 南京,210044

2 南京信息工程大学 信息与控制学院,南京, 210044

越高.但是,软件网络一般都是加权有向网络,单单从“度”上来衡量类的重要性并不准确.例如,对于一个具有特殊功能的类,本身它的度并不大,但是若将其从软件中去除就可能直接导致该软件不可运转.

通过对无标度网络的研究知道,一般节点发生缺陷的情况有两种:一种是随机节点设计缺陷,也就是软件网络中存在一部分随机节点具有设计缺陷,但这部分节点的功能缺失一般并不影响软件整体的正常运转;二是重要度节点的设计缺陷,即软件网络中节点重要度比较高的节点有设计缺陷.在软件网络中,由于度分布呈现幂率分布^[7-8]的非均匀特性,这使得这类节点被调用的次数很高,而且它们占软件网络中所有节点个数不到5%^[9],却实现了软件最主要的功能.因此,一旦这些类的功能出现缺失,有可能就会直接导致系统崩溃.鉴于此,如果在软件开发初期和软件测试^[10]过程中对这些极少数的节点重点对待的话,那么不仅可以缩短软件开发测试周期,而且还能使软件的质量有所提高.因此,如何更有效且准确地找到这些重要的节点便是当前要做的工作.

2 软件网络中的节点重要度方法

Method of node importance in software networks

在复杂网络中,通常以介数或度来衡量节点和边的重要性.介数虽能很好地反映相应节点在当前网络中的影响力,但其仍存在的问题,且在庞大的复杂系统中,计算相当复杂.同样,节点的连接度只能在一定程度上反映节点在网络中的重要性,并不能客观反映该节点对整个网络的影响力,但通过连接度可以评价它的构造复杂性.在此结合网络凝聚度^[11]的定义,并根据软件网络的特性,引入软件网络节点重要度的概念.以此作为软件网络查找重要节点的指标.

首先,对于软件网络,通常将复杂软件系统中不同标度上可重用的元素(类、对象等)视为节点,节点之间的相互关系(继承、调用等)视为边,以此抽象成软件网络模型.在面向对象的软件系统中,将类视为节点,以 V_i (i 表示软件网络中的某个节点)表示.类与类之间的继承、调用关系视为边,为体现软件中类之间的调用关系,这里使用边加权值,以 W_{ij} (表示从节点 V_i 到节点 V_j 的边权值)表示权值.此外,因为软件网络是一个特殊的复杂网络,类之间存在着调用和被调用的关系,所以相应的节点之间的连接度对应着也会有入度、出度2个方向;有关软件

网络中具体节点之间边的连接方向确定可参照表(1)所示.

表1 软件网络中边方向的确定

Table 1 Determination of side direction in software networks

类之间的关系	相互作用的节点	边的方向
调用	节点 V_i 调用节点 V_j 的方法或成员	$V_i \rightarrow V_j$
继承	节点 V_i 继承节点 V_j 的方法或成员	$V_i \rightarrow V_j$

根据以上有关软件网络的节点、边定义,在此定义面向对象软件系统的软件网络模型 T 由节点集 $V = \{V_1, V_2, V_3, \dots, V_i, \dots, V_n\}$ ($0 < i \leq n$)和加权有向边集 $W = \{W_{11}, W_{12}, W_{13}, \dots, W_{ij}, \dots, W_{nn}\}$ ($0 < i, j \leq n$)组成.表示为 $T(V, W)$.

复杂网络中,网络凝聚度用以表征1个节点在与其相连的节点收缩成1个新的节点后,引起整个网络凝聚度的变化程度.例如将星型网络的中心节点进行收缩组成1个节点团后,整个网络只剩1个节点,此时网络的凝聚度变化最大.而对其他节点进行收缩后并不能引起凝聚度有太大的变化.因此可以认为网络中的节点收缩后,引起网络凝聚度变化较大的节点,其重要程度相对要高.

其中影响凝聚度变化的因素有表示网络规模的节点数 n ,及评价节点之间连通能力的平均最短路径长度 l .目前已有研究证明,在复杂网络中,网络凝聚度能较好表示节点的重要性.但是鉴于软件网络是1个加权有向图,以上2个参数并不能完全描述其特殊性.为此,在计算平均路径长度 l 表示各节点对之间不同的复杂度的时候,引入边加权值 W ,以此更恰当地反映路径起点到终点之间的复杂度衡量.定义软件网络凝聚度如下:

定义1

$$\partial(T) = \frac{1}{n \cdot l} = \frac{1}{2 \cdot \sum_{j \in N_i} \frac{1}{W_{ij}}} = \frac{n-1}{2 \cdot \sum_{j \in N_i} \frac{1}{W_{ij}}}. \quad (1)$$

式(1)中: $n(n > 1)$ 为软件网络 T 的节点个数; l 表示软件网络 T 中每2个节点间的平均路径长度;矩阵元 W_{ij} ($W_{ij} > 0$)代表相邻2点间的边权, W_{ij} 的值越大,说明2个节点间联系越紧密,体现在加权网络中,即为 $l_{ij} = l/W_{ij}$;当网络中只有2个节点,即 $n = 1$ 时,网络的凝聚度最高,此时 $\partial(T) = 1$.

另外,因为软件网络中所采用的权值为相似权,所以对于节点收缩后的节点团的权值,不可直接相

加得到,在此使用调和平均值来求得,其公式如下:

$$l_{ik} = \frac{W_{ij}W_{jk}}{W_{ij} + W_{jk}}. \quad (2)$$

在已知凝聚度,对网络中各节点进行收缩后计算节点重要度的时候,仍需考虑软件网络的特殊性.因为软件网络中节点的入度和出度方向在实际的软件应用过程中体现着完全不同的作用.出度方向更能表征2个节点对其他节点的依赖性,从而隐含着较大的潜在出错概率,因而显得更为重要.故在计算节点重要度时,应着重考虑当前节点在出度方向上的因素.定义节点重要度如下:

定义 2

$$M_l(V_i) = \frac{k_v^{\text{out}}}{k_{\text{max}}^{\text{out}}} \cdot \left(1 - \frac{\partial[T]}{\partial[T \times V_i]}\right) = \frac{k_v^{\text{out}}}{k_{\text{max}}^{\text{out}}} \cdot \left(1 - \frac{(n - k_i) \cdot l(T \times V_i)}{n \cdot l(T)}\right). \quad (3)$$

式(3)中, $\frac{k_v^{\text{out}}}{k_{\text{max}}^{\text{out}}}$ 为软件网络的出度系数,其中, k_v^{out} 表

示当前节点的出度值,若当前节点 k_v^{out} 出度为0,则设其值为1; $k_{\text{max}}^{\text{out}}$ 表示软件网络中节点的最大出度值.此外, $1 \leq l(T \times V_i) < l(T)$, $k_i \geq 1$, $T \times V_i$ 表示将节点 V_i 收缩后所得到的网络图.节点重要度表征通过对软件网络中各节点的节点重要度进行计算比较,以找到软件中结构复杂,在功能上又相对比较重要的类,从而有利于快速解决软件中存在的缺陷,并提高软件开发质量.其中, $M_l(V_i)$ 的值越大,对应的节点就越重要.

3 实例分析

Analysis of examples

作为目前流行的面向对象编程语言,Java 程序广泛应用于各种嵌入式系统、台式机和服务器上.本文选取其中一种较具代表性的具有开放源代码的软件系统:Tomcat.将其简化的类图抽象为软件网络有向加权图,如图1所示.

在图1中,各节点类之间的调用关系如下: V_1 派生于 V_4 ; V_2 调用 V_3 的方法; V_3 由 V_6 实现; V_4 有一个连接对象 V_1 和应用逻辑类 V_2 ; V_5 从 V_7 派生,包括 V_4 类对象; V_6 从 V_5 派生,实现 V_3 ,并调用核心对象 V_8 的服务方法; V_7 是所有监听器的基类; V_8 为系统的核心控制对象.

如图1所示,该软件网络中有8个节点,10条边.显然,节点的连接度不能作为衡量节点重要度的

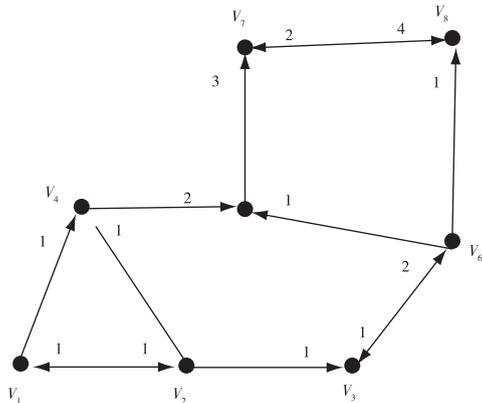


图1 Tomcat 软件简化的类图

Fig. 1 Simplified class diagram of Tomcat software

标准.其中,节点 V_2, V_6 的出度为3, V_1 的出度为2;但是从实际的软件模型的重要性分析上来看,它们在网络中的重要度是有区别的.在此,采用节点团方式将各节点进行收缩,并计算它们的重要度.其评估结果如表2所示.

表2 节点重要度评估结果

Table 2 Results of the evaluation of node importance

节点	连接度(出度)	点权值	介数	节点重要度
V_1	2(2)	1.00	0.228 6	0.211 3
V_2	3(3)	1.00	0.213 4	0.231 6
V_3	2(1)	0.50	0.109 3	0.114 1
V_4	3(1)	0.50	0.121 6	0.120 8
V_5	3(1)	0.67	0.128 7	0.129 5
V_6	3(3)	2.00	0.321 5	0.314 3
V_7	2(1)	1.20	0.168 7	0.159 6
V_8	2(1)	0.80	0.138 1	0.144 0

由表2可见,各节点按重要度排序依次为 $V_6 \rightarrow V_2 \rightarrow V_1 \rightarrow V_7 \rightarrow V_8 \rightarrow V_5 \rightarrow V_4, V_3$.该方法与介数法的排序结果是一致的,而且其结果与根据类的具体含义在软件中的重要度所得出的结果也是基本一致的,节点重要度越大的节点越重要.

4 结束语

Concluding remarks

研究软件节点重要度的度量方法,对发展计算型软件工程有重要意义.本文借鉴复杂网络的思想来考察软件系统的整体性质,不仅能动态地反映出软件系统的网络特性,而且易于发现其系统特征规律.但是,目前基于复杂网络的软件研究仅仅还是停

留在对某些静态特性进行度量研究的水平上,还有很多方面,比如软件节点重要度随着软件模型的动态增长而改变,其特征和变化规律等还有待于进一步研究.

参考文献

References

- [1] Mechelle G, Yong K, David G. The Vital Few Versus the Trivial Many; Examining the Pareto Principle for Software [R]. 29th Annual International Computer Software and Applications Conference (COMPSAC05), Volume I. Edinburgh, Scotland, UK: COMPSAC'05, 2005:179-185
- [2] Malaya Y, Denton J. Module size distribution and defect density [R]. Proc of the 11th Int'l Symp, on Software Reliability Engineering. New York: IEEE Computer Society Press, 2000:62-71
- [3] Chidamber S R, Kemerer C F. A metrics suite for object oriented design [J]. IEEE Transactions on Software Engineering, 1994, 20 (6):476-492
- [4] Abreu F. The MOOD metrics Set [R]. In Proceedings of ECOOP'95 Workshop on Metrics, Aarhus, DenMark, 1995:90-99
- [5] 韩明畅, 李德毅, 刘常昱, 等. 软件中的网络化特征及其对软件质量的贡献 [J]. 计算机工程与应用, 2006, 20(3):9-10
HAN Mingchang, LI Deyi, LIU Changyu, et al. Network-oriented

- features in software and their contributions to the quality of software [J]. Computer Engineering and Applications, 2006, 20(3):9-10
- [6] 汪小凡. 复杂网络理论及其应用 [M]. 北京: 清华大学出版社, 2006:106-136
WANG Xiaofan. Theory of complex networks and its application [M]. Beijing: Tsinghua University Press, 2006:106-136
- [7] Sole R V, Ferrer R, Montoya J M, et al. Tinkering and emergence in complex networks [J]. Complexity, 2002, 8(1):20-33
- [8] 李梦辉, 樊瑛, 狄增如. 加权网络 [C] // 郭雷, 许晓鸣. 复杂网络. 上海: 上海科技教育出版社, 2006:27-31
LI Menghui, FAN Ying, DI Zengru. Weighted networks [C] // GUO Lei, XU Xiaoming. Complex network. Shanghai: Press of Shanghai Science and Technology Education, 2006:27-31
- [9] Albert R, Jeong H, Bumbasi A L. Error and attack tolerance of complex networks [J]. Nature, 2000(4):406-413
- [10] Elfriede D. 有效软件测试 [M]. 北京: 中国电力出版社, 2004:130-189
- [11] 谭跃进, 吴俊, 邓宏钟. 复杂网络中节点重要度评估的节点收缩方法 [J]. 系统工程理论与实践, 2006, 11(5):79-85
TAN Yuejin, WU Jun, DENG Hongzhong. A contraction method of the node importance assessment in complex networks [J]. System Engineering Theory and Practice, 2006, 11(5):79-85

Research on software measurement based on complex networks

MA Jieliang¹ ZHENG Weicai² WANG Junchao²

1 School of Electronic & Information Engineering, Nanjing University of Information Science & Technology, Nanjing 210044

2 School of Information Science & Technology, Nanjing University of Information Science & Technology, Nanjing 210044

Abstract With the augmentation of the scale of software systems, the interaction between the software elements has become increasingly complicated, which leads the development of software to be instable, making problems in software quality gradually prominent. Based on the achievements obtained in complex networks, the software structure is made into a graph of weighted directed networks; then, defining the Out-degree coefficient by a corresponding index of node important degree, to find and judge the classes with high defect probability, the flaw occurrence characteristics of scale-free networks are utilized to study the defect distribution of software structure. The last results show that the measurement method is effective in checking for software defective classes, which is helpful to finding out defects in the early phase of developing software and reducing the cost of software development and testing so as to improve the quality of software developed.

Key words scale-free network; node importance; node mass; software model