

基于复杂网络的软件复杂性度量研究

李 兵^{1,2}, 王 浩², 李增扬¹, 何克清¹, 余敦辉^{1,2}

(1. 武汉大学软件工程国家重点实验室, 湖北武汉 430072; 2. 湖北大学数学与计算机科学学院, 湖北武汉 430062)

摘 要: 软件开发者对于日趋复杂的软件系统的理解和控制越来越困难, 传统软件工程正接近其复杂性和可扩展性的极限. 复杂性使软件开发困难, 质量难以保证. 复杂网络理论的最新研究成果, 为软件复杂性度量提供了新的数学基础. 讨论了软件复杂性的形成原因和度量方法, 介绍了目前复杂网络与软件复杂性结合的研究工作. 探讨了基于复杂网络的软件结构复杂性度量方法, 提出一种结合复杂网络和演化算法的软件演化复杂性度量模型.

关键词: 软件复杂性; 复杂网络; 软件度量; 下一代软件工程

中图分类号: TP311.5 **文献标识码:** A **文章编号:** 0372-2112 (2006) 12A-2371-05

Software Complexity Metrics Based on Complex Networks

LI Bing^{1,2}, WANG Hao², LI Zheng-yang¹, HE Ke-qing¹, YU Dun-hui^{1,2}

(1. The State Key Laboratory of Software Engineering, Wuhan University, Wuhan, Hubei 430072, China;

2. Faculty of Mathematics and Computer Science, Hubei University, Wuhan, Hubei 430062, China)

Abstract: Human understanding and control of complex systems development often seem unattainable goals. The problem is substantial for future construction of network-centric, large-scale software systems. Evidence suggests that software engineering is reaching complexity and scalability limits of technologies. Software development and software quality is troubled with software complexity. New and emerging research efforts on complex networks introduce a new mathematical foundation for software complexity metrics. This paper discusses the forming mechanism and metrics methods of software complexity, and introduces recent progresses about associating software complexity with complex networks. Based on software complex networks, some methods of software structure complexity metrics are proposed. Focusing on combination of complex networks model and evolutionary algorithm, a model for software evolutionary complexity metrics is presented.

Key words: software complexity; complex networks; complexity Metrics; next-generation software engineering (NGSE)

1 引言

生成高质量的软件是所有的软件工程方法努力实现的目标. 然而, 软件系统的设计开发却总不能尽如人意, 最主要的原因是软件系统的复杂程度越来越高. 越复杂的软件就越难以保证其质量、费用和生产率, 软件开发就会经常处于失控状态. 软件中一个极小的错误可能带来灾难性的后果, 甚至形成一种“雪崩效应”——大型软件系统的无数细节和可能都会构成其复杂度的质变. 因此, 目前兴起的下一代软件工程 (NGSE, Next-Generation Software Engineering) 正致力于将软件工程专业发展为一个“计算型”的工程学科, 基于数学理论和语义导向, 在软件系统设计开发过程中, 实现高可靠、低成本和自动化的智能控制.

软件度量是使软件设计与开发的科学性的重要保证, 软

件开发之所以经常处于失控状态的原因正是缺乏合理的度量. 软件复杂性的度量又与软件的可靠性、可维护性等质量因素密切相关. 在 60 年代, 软件度量的基础性工作就已展开^[1,2]. McCabe 和 Halstead 在 70 年代中期提出的 McCabe 环复杂度是一种较为成功的结构化程序度量方法^[3]. 它通过计算线性独立路径条数来度量程序的复杂性, 值得注意的是, 它首次考虑了软件内部的网络拓扑特性. 对于软件度量本身性质的评价, 比较著名的有 Brito 针对面向对象的软件生命周期和质量保证的 7 条标准^[4], 以及 Weyuker 提出的一组形式化评估软件度量性质的定理^[5].

在 80 年代末, 面向对象 (OO, Object-Oriented) 思想和方法成为软件设计与开发的主流, 但上述传统的度量方法无法应用于许多面向对象的特性. 因此, 出现了一些适合面向对象软件的程度方法. 较为有代表性的是 Chidamber 和 Kemerer 提出

收稿日期: 2006-09-22; 修回日期 2006-11-10

基金项目: 国家 973 计划前期研究专项 (No. 2006CB708302); 国家自然科学基金 (No. 90604005); “十五”国家重大科技专项 (No. 2002BA906A21-25); 湖北省自然科学基金 (No. 2005ABA123, No. 2005ABA240, No. 2006ABA228); 软件工程国家重点实验室开放基金 (No. SKLSE05-07, No. SKLSE05-19); 国家 863 高技术研究发展计划 (No. 2006AA04Z156)

的基于继承树的 C&K 度量方法^[6],以及 1995 年 Brito 和 Abreu 提出的 MOOD 方法. 这些方法主要聚焦于:(1)类的内部复杂性(如类的加权方法数、类内聚缺乏度、方法隐藏因子、属性隐藏因子);(2)类之间继承的复杂性(继承树深度、继承方法数比例);(3)类之间的耦合复杂性(对象类之间的耦合数、所有可能的关联类对的数目). 上述方法的问题在于:(1)仅注重统计与对象类有关的数量(规模),虽然注意了对象类之间的耦合,但也是一种数量的统计,不能体现出类之间的相互作用;(2)侧重微观上的统计,缺乏全局和总体上的复杂性测量,而系统的复杂性通常是整体上的表现;(3)各度量之间存在相关性的内在依据;(4)仍停留在软件内部属性的度量阶段,缺乏外部属性和内部属性之间关系的确定性定义.

有鉴于此,本文根据目前新兴的复杂网络研究成果,结合面向对象软件的内部拓扑结构特点,提出了一种能从整体上对软件复杂性进行度量的模型,同时尝试对传统软件复杂性度量进行统合.

2 复杂网络与软件网络观

2.1 复杂网络研究的进展

系统的复杂性主要决定于元素之间的交互,只要能保持系统元素之间交互的基本性质,那么系统的基本特性就不会改变. 尽管组成各种网络的元素不同、各类实际网络有其自身的复杂性,但却存在某些代表普遍规律的共同性质. 因此,网络模型是描述复杂系统的最有效模型,目前大量最新研究发现:网络的拓扑结构决定着网络所拥有的特性.

长期以来,由于对网络的拓扑结构性知之甚少,在以前的研究中,科学家们往往忽略了网络的拓扑性质. 在最近 40 多年的时间里,由 Erdős 和 Alfréd Rényi 开创的随机图理论一直被奉为研究网络的经典模型. 直到最近几年,由于计算机数据处理和运算能力的飞速发展,科学家们发现大量的、真实系统的网络模型既不是规则网络,也不是随机网络,而是呈现出出乎意料的统计特征,由于这样的网络是真实复杂系统的拓扑抽象,因此被称为复杂网络(complex networks). 研究表明,规则网络具有大的集聚系数和大的平均距离,随机网络具有小的集聚系数和小的平均距离,随机网络连接度分布为钟形曲线分布且峰值取平均值,每个节点有大致相同数目的连接度. 1998 年 Watts 和 Strogatz 在《Nature》杂志上发表文章,表明大量真实网络都具有小世界效应(SW, Small-World effect),即具有大的集聚系数和小的平均距离的网络^[7]. 1999 年 Barabasi 和 Albert 在《Science》上发表文章,指出许多现实世界中的复杂网络的连接度分布服从幂律分布^[9]. 由于幂律分布的网络不像随机网络和规则网络可以将平均度看作节点度的特征标度,所以将这类网络称为无标度(SF, Scale-Free)网络. SF 特性刻画了复杂网络的不均匀复杂性,即大部分节点只有少数连结,而少数节点则拥有大量的连结. Barabasi 和 Albert 进一步提出导致 SF 特性的 BA 模型,即 SF 网络的演化主要包括两个部分:第一,增长性. 网络的规模随着时间在不断地扩大;第二,优先连接. 新增加的节点与网络中节点 i 的连接概率为 $P(k_i) = k_i / \sum k_j$, 其中 k_i 为节点 i 的度数. BA 模型中,众多节点

优先连接到高连接度的节点上,呈现一种“富者逾富”的现象.

2.2 软件系统的网络观

大型复杂软件是将复杂的问题分解为多个部分,再由多个开发者共同完成. 在开发过程中,根据功能需要将复杂的功能分解为大量可重用的元素(类、对象、子程序、构件等),若将这些元素视为节点,这些节点之间的相互关系就构成一张复杂的协作关系网络. 最近,对这种软件复杂网络研究表明,大型软件系统内部结构并不是随机的,也具有小世界效应和无标度特性^[10~12]. 实际上,很多软件设计工具都利用图示语言来表示这些软件构件以及它们的关系,如面向对象程序设计的统一建模语言 UML(Unified Modeling Language),有标准的规则集合用于图的概念、设计和实现,只不过并未考虑图的全面组织. 如果将每个节点代表一个类或构件,类或构件的关系表示为节点之间的边,就可以建立大型软件的复杂网络模型.

从软件设计范型的发展来看(图 1),早期自由设计的软件规模较小,同时也没有公认的规范,受程序设计者的个人影响较大,可以视为某种随机网络模型. 而强调结构化设计的软件,出现模块化的思想和规范,程序的控制结构也趋于严格,将这种软件网络化后,可以发现大量较规则的拓扑结构,同时,模块之间有较少长程连接,如调用语句、Goto 类语句等,可视为 SW 或 SF 模型;而面向对象设计的软件,更接近于客观世界,可扩展性和开放性更强, SF 模型的特性更突出. 目前的基于构件的设计,有大量的层次化特征. 其原因主要是构件生产的自治性和异构性,同时,结合最近的 Web 服务技术形成的面向服务的体系结构(SOA, Service-Oriented Architecture),软构件以服务的形式相联系,彼此的耦合更松散,软件系统的网络化、服务化趋势将越来越明显.

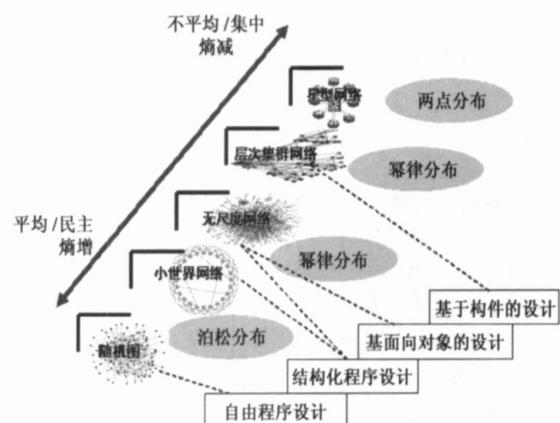


图 1 软件设计范型与复杂网络

因此,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. 从复杂网络的角度,在更加坚实的数学理论基础,重新认识软件以及软件工程,将是未来“计算型”软件工程发展的总体趋向.

3 基于复杂网络的软件复杂性度量

3.1 软件系统的复杂网络建模

对于软件开发不同阶段的不同元素,都可以建立网络模型,如 Valverde 运用无向图的模型对 Java Development Frame-

work 1.2(JDK1.2)的类图网络进行研究,发现了其中存在的小世界和无标度特性^[10]. Potanin 等对几个 OO 程序中对象图的结构进行了研究,分析了其中的对象动态交互时的一些拓扑性质^[11]. Wheeldon 等对包括 JDK 在内的几个软件的继承关系、接口关系的网络分别进行分析,发现了其中的一些幂律关系.

目前,对于软件系统的复杂网络研究主要是对已有软件采用逆向工程方法得出其中的组织结构进行分析,发现其中的复杂网络特性.这些特性无疑反映出软件系统的一些整体度量性质,但是对于这些特性的形成机制和演化过程描述并不清楚,同时主要是采用从代码出发的“反向”研究途径,没有真正涉及软件系统开发设计所遵循的组织结构.更为重要的是,所研究的度分布、平均距离等参数仅反映出结构的拓扑特征,没有结合到更具体的软件度量.

我们认为,按照 Brito 的“度量应在生命周期的早期可用”标准,应针对设计开发阶段的软件结构进行建模.由于 UML 是目前软件建模的事实标准,因此,我们给出了一个针对 UML 类图元素建立软件复杂网络模型例子.

定义 1 任何 UML 类图可以表示为一个软件网络 $G = (V, E)$. G 是一个连通图,包括一个节点集 V 和一个边集 E .
软件网络 = 节点 | 边

定义 2 软件网络的节点与 UML 中类相对应. 节点 = UML 类

定义 3 G 中每个节点的复杂度 VF (Vertex Factor) 由一个复杂性因子相关的函数来表达. $VF(i) = f(x_j), i \in [1, N], j \in [1, M], N$ 为 G 中最大节点数, M 为复杂性因子个数.

定义 4 E 中的边对应类之间的关系. 有序对 V_i, V_j 表示一条 V_i 为起点, V_j 为终点的边. 边的方向确定原则如表 1 所示. 根据具体情况, V_i, V_j 可以赋予一个权重 w_{ij} .

表 1 软件网络中边方向的确定

类之间的关系	边的方向	
泛化 (Generalization)	$V_j \xrightarrow{\quad} V_i$	V_i, V_j
依赖 (Dependency)	$V_i \dashrightarrow V_j$	V_i, V_j
关联 (Association)	$V_i \xrightarrow{\quad} V_j$	V_i, V_j, V_j, V_i
聚合 (Aggregation)	$V_j \xrightarrow{\quad} V_i$	V_j, V_i
合成 (Composition)	$V_j \xrightarrow{\quad} V_i$	V_j, V_i

在进行具体分析时,如果根据需要放宽限制,在忽略定义 4 的情况下即是无向图的情况.

3.2 基于复杂网络的软件结构复杂性度量

上述无权网络只能给出顶点之间的相互作用存在与否的定性描述,当顶点的特性以及顶点之间的相互作用强度的差异起着至关重要的作用时,无权网的描述就不够了,节点和边的权重将提供更加细致的定量刻画.通过将度量要素以权重的形式赋予节点和边,就可以定量地刻画和分析软件的结构和行为,以便对软件系统更好地理解、评估、预测、控制和改进.因此,软件网络模型可以进行的结构复杂性度量有如下类型:

- (1) 全局拓扑特性的度量:

平均距离 D (average distance). $d_{\min}(i, j)$ 是节点 i, j 之间至少要经过的边数. 对于无向网络,网络 G 的平均距离定义为:

$$D = \frac{1}{N(N-1)} \sum_{i,j \in V} d_{\min}(i, j)$$

其中, N 表示网络的节点数, V 表示网络的节点集合.

网络集聚系数 C (clustering coefficient). 专门用来度量网络节点聚集度的情况. 某个节点 i 的聚集度定义为 i 的相邻节点间实际存在的边数比上节点 i 的相邻节点间完全连接的边数. 节点 i 的集聚系数为:

$$C(i) = \frac{2E(i)}{k_i(k_i - 1)}$$

整个网络的集聚系数是每个节点集聚系数的平均值:

$$C_G = \frac{1}{N} \sum_{i=1}^N C(i)$$

度和度分布 (degree & degree-distribution). 在有向网络中,节点的度包括出度和入度. 无向网络中没有出度和入度的区别. 每个节点的度能够描述该节点的重要程度,用 $P(k)$ 表示网络中度数为 k 的节点个数或在网络中所占的比例,网络的 $P(k)$ 函数体现了该网络的整体特性.

(2) 节点的复杂性度量

节点的复杂性可能与多个功能因子有关,因此定义 3 中建议采用一个这些功能因子的函数来表达. 如 C&K 模型中的类方法的带权和 (WMC, Weighted Methods Per Class). 如果类 C 定义了 n 个方法 M_1, \dots, M_n , 设 c_1, \dots, c_n 是这些方法的复杂性, 则 $WMC = \sum_{i=1}^n C_i$, 简单情况下, WMC 为该类的方法数. 其他的因素,包括 C&K 模型中的类方法内聚度缺乏 (LCOM, Lack of Cohesion in Methods) 等与类本身相关的因子都可以结合到一个函数中共同表达类的复杂度,该函数的构成需要根据经验数据进行分析获得.

(3) 交互的复杂性度量

总体上的交互复杂性可以通过度分布刻画,如入度大的节点(类)说明其重用度大,而出度大的节点(类)往往比较复杂. 在 C&K 模型中有几个度量与类之间的交互相关,且都可以在本模型中得到解释. 如 C&K 模型中的继承树深度 (DIT, Depth of Inheritance Tree) 可以用节点之间的距离来刻画. 子类数目 (NOC, Number of Children)、对象类的耦合度 (COB, Coupling between Object Classes) 和类响应数目 (RFC, Response For a Class) 都可以用与节点的度有关的测量来描述. 同时,在软件运行过程中的交互强度等因素也直接关系到软件的复杂性,这可以赋予网络中的边以权重来度量.

(4) 聚集复杂性度量

现有的研究表明,网络的集团结构 (Community Structure) 是指节点之间的连接程度不同而形成的“抱团”结构. 在集团的内部,节点之间的连接程度明显高于不同集团之间的节点之间的连接程度,集团结构是反映网络结构整体性质的重要特征. 在软件协作网络中,节点的相互作用范围往往就是节点内在语义协同的成果,如包、构件等就是将许多类集成一个更高层次的单位,形成一个高内聚、低耦合的类的集合. 这种

- gram quality[A]. Proceedings of the 23rd ACM National Conference[C]. New York:ACM Press,1968. 671 - 677.
- [2] Halstead M H. Elements of Software Science[M]. New York: Elsevier North-Holland,1977.
- [3] McCabe T. A complexity measure[J]. IEEE Transactions on Software Engineering,1976,2(4):308 - 320.
- [4] Brito F, Abreu E, MOOD-metrics for object-oriented design [A]. OOPSLA '94 Workshop on Pragmatic and Theoretical Directions in Object-Oriented Software Metrics [C]. Portland: OR,1994.
- [5] Weyuker E. Evaluating software complexity measures[J]. IEEE Transactions on Software Engineering,1988,14:1357 - 1365.
- [6] Chidamber S R, Kemerer C F. A metrics suite for object oriented design [J]. IEEE Transactions on Software Engineering, 1994,20(6):476 - 492.
- [7] Watts D J, Strogatz S H. Collective dynamics of small-world networks[J]. Nature,1998,393:440 - 442.
- [8] Batabasi A L and Albert R. Emergence of scaling in random networks[J]. Science,1999,286:509 - 512.
- [9] Valverde S, Ferrer Cancho R, Sole R V. Scale-free networks from optimal design[J]. Europhysics Letters,2002,60:512 - 517.
- [10] Sole R V, Ferrer R, Montoya J M, Valverde S. Tinkering and emergence in complex networks[J]. Complexity,2002,8(1): 20 - 33.
- [11] A Potanin, J Noble, M Frean, Robert Biddle. Scale-free geometry in object-oriented programs [J]. Communications of the ACM,2005,48(5):99 - 103.
- [12] Liu Bin, Li Deyi, Li Bing. Mining representative nodes in scale-free networks[J]. Dynamics of Continuous, Discrete and Impulsive Systems (Series B: Applications and Algorithms), 2006,13(3):395 - 400.
- [13] Valverde S, Solé R. Hierarchical Small-Worlds in Software Architecture[R]. Santa Fe Institute working paper, SFI/03-07-044,2003.

作者简介:



李 兵 男,1969 年生于武汉,武汉大学软件工程国家重点实验室副教授.研究方向为软件工程、系统集成和复杂网络.

E-mail:libing@sklse.org

王 浩 男,1980 年生于湖北,硕士研究生.研究方向为软件工程和复杂网络.

李增扬 男,1981 年生于江西,硕士研究生.研究方向为软件工程和复杂网络.

何克清 男,1947 年生于湖北,武汉大学软件工程国家重点实验室教授.研究方向为软件工程.

余敦辉 男,1979 年生于湖北,博士研究生.研究方向为软件工程和复杂网络.