# 软件网络分形结构特征研究

潘 浩[1] 郑 巍[1,2] 张紫枫[1] 芦超群[1]
(南昌航空大学软件学院 南昌 330063)[1] (西安电子科技大学数学与统计学院 西安 710126)[2]

**摘 要** 随着互联网技术的发展,软件系统的规模也在日益扩大,软件功能的变化使得软件网络的结构呈现差异化。软件网络的分形结构体现出软件网络整体和局部对象类之间依赖调用的自相似性,可以从代码层面上对软件系统结构以及功能进行分析。文中对软件网络分形结构特征进行研究:首先,综合类之间的依赖关系,对软件网络进行加权;然后利用基于网络中心性的盒子算法计算加权软件网络结构的分形维度,以分析其分形结构特征;最后,利用上述方法对spring和struts2这两款具有代表性的java软件框架进行分析。实验表明,这两款软件以及它们的子模块都具有分形结构特征,软件网络的分形维数随着模块功能复杂程度的增加而增加,功能综合性较强的软件网络分形维数要大于功能专一的软件网络;并且在版本演化过程中,软件网络的分形维数随着软件功能的丰富也呈现上升的趋势。

**关键词** 软件网络,分形,软件系统结构,软件演化
**中图法分类号** TP311.5 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2019.02.026

## Study on Fractal Features of Software Networks

PAN Hao[1] ZHENG Wei[1,2] ZHANG Zi-feng[1] LU Chao-qun[1]
(School of Software, Nanchang Hangkong University, Nanchang 330063, China)[1]
(School of Mathematics and Statistics, Xidian University, Xi'an 710126, China)[2]

**Abstract** With the development of the Internet technology, the scale of software architecture is growing beyond the requirements, and the software function changes the structure of software network. The fractal structure of the software network reflects the self-similarity of the modules and the whole software network, and can analyze the architecture of software from the code level. This paper researched the fractal characteristic of software networks. First, the software networks are weighted by the complex relationships between the classes. Further more, a network centrality based box algorithm is utilized to calculate the fractal dimension of the software network. At last, two representative software of spring and struts2 are analyzed through experiments, and the results show that the both two framework and their modules have fractal features and the fractal dimension of the software network increases with the complexity of the module. The fractal dimension of the software network with more comprehensive functions is bigger than that of the software network with special functions. Also the fractal dimension increases with the evolution of the software version in which the software function is gradually improved.

**Keywords** Software network, Fractal, Architecture of software, Software evolution

## 1 引言

随着互联网的发展,软件的规模逐渐扩大,面向对象技术在软件的开发中得到了广泛的应用。在系统结构、模块生命周期以及项目维护方面,基于面向对象技术的软件开发与面向结构技术有着显著的区别。面向对象软件开发的实践表明,软件规模的扩大会造成软件模块以及对象的增加,对象之间依赖关系的复杂性会影响软件系统的质量。软件的结构对软件的质量具有非常深远的影响,正如Torres在文献[1]中所指出的:软件的结构是软件容错的基础。近年来的研究表明,通过软件基本组成元素及其之间的相互关系可构建大规模复杂软件系统的软件内部结构网络,研究软件内部结构潜在的客观特性及其度量方式,构建软件网络模型并探索其固有的科学特性,是对软件系统复杂性进行深入研究的基础[2]。

大规模的软件系统属于复杂系统,利用软件的组成单元和它们之间的依赖关系组成的网络称为软件网络[3]。通常,一个功能完整、性能优良的大型软件网络是一个具有幂率分布的无标度网络,具有小世界特征[4],这样的特征使得软件网络的节点之间具有很小的平均最短路径以及节点连接偏好性[5-6]。高洋等[7]借助复杂网络的理论和方法引入了软件结构的度量方法,并通过实验表明软件网络中度分布和点权分布均存在一定程度的指数衰减现象。Chong等[8]利用加权网

到稿日期:2018-08-07 返修日期:2018-09-13 本文受国家自然科学基金项目(61501217,61762065)资助。

**潘 浩**(1993-),男,硕士生,主要研究方向为软件工程、复杂网络,E-mail:hetto0224@outlook.com;**郑 巍**(1982-),男,博士,副教授,硕士生导师,主要研究方向为复杂网络、网络优化、软件工程、移动社交网络,E-mail:zhengwei_nchu@126.com(通信作者);**张紫枫**(1995-),女,硕士生,主要研究方向为软件工程、复杂网络;**芦超群**(1997-),男,主要研究方向为软件工程、复杂网络。

络对面向对象软件系统进行网络化，并捕获网络结构的特征来定义软件系统的可维护性以及可靠性。汪北阳等[9]利用复杂网络理论，分析了各统计特性与软件网络结点影响的关系，进一步提出了软件网络关键结点的概念，并通过仿真实验进行了验证。上述研究借助复杂网络理论研究软件系统体系结构，对面向对象软件中对象之间复杂性进行了具体描述，提出了软件网络构建方法；并且在软件网络基础之上，分析了重要节点即重要对象类在软件中的作用。软件网络的无标度特性可以解释为关键对象类的重用程度，或者对象类之间联系的程度分布具有无标度特征。

复杂网络系统中的分形结构研究是系统网络演化的一种重要的结构特征，对网络构成以及演化的预测和解释有着重要的意义[10]。在软件网络中，分形维数和面向对象软件质量有着重要的联系。Concas 等分析了软件版本演化过程中分形维数和 CBO(类间耦合)、RFC(类响应)等参数的变化，并发现分形维数与 CK 度量集中的 CBO 和 RFC 等度量参数变化保持一致的线性变化，并将分形维数作为软件质量评估参数[11-12]。文献[13]研究了分形维数和软件缺陷的关系，并通过实验验证了这种关系的存在性。上述研究对面向对象软件进行软件网络构建，并使用盒数法计算分形维数。而对象类之间关系的复杂性是软件网络构建的重要因素，软件网络的边并不能完全表示对象类之间的关系。

除了软件系统的整体演化规律，面向对象软件系统中模块的设计也是影响软件质量的重要因素。在面向对象软件结构中，一个模块内的各个对象类是紧密相连的，对象类的内聚表示对象类紧密相连的程度。随着软件系统规模的扩大，软件模块的合理设计有利于软件工程后期的维护和升级。因此，在上述研究背景下，文中对类之间关系的复杂性进行合理的定义，并对多模块面向对象软件系统进行软件网络构建。由于盒数法中盒子的选择具有随机性，而多模块面向对象软件中的节点之间具有复杂的依赖关系，导致网络中的盒子覆盖具有一定的随机性，且破坏了软件对象类之间的连通性。文中利用基于节点中心性盒数法计算加权软件网络下的分形维数，并对 spring 以及 struts2 软件网络及其重要子模块软件网络进行分形结构特征的测算。实验表明，这两款软件网络及其子模块软件网络都具有分形结构特征，通过对其子模块的功能进行研究，发现模块的功能设计会影响其对象类之间的依赖关系，功能专一的模块，其依赖关系比较简单，网络的覆盖能力与功能复杂的模块相比会有所降低。此外，对软件版本演化和分形结构特征进行了分析，发现随着版本的迭代以及功能的完善，软件网络分形结构特征一直存在，且功能的完善使得软件网络覆盖能力增加，其分形维数也随之增加。通过对软件网络结构的研究，可以在代码层面上对软件系统进行分析，为大型软件系统的设计与评估提供新的思路。

## 2　研究方法

### 2.1　软件网络的构建

在软件开发过程中，复杂的功能系统可以分解为对象类单元集合，对象类之间的依赖和交互可以实现软件系统的各项功能。在软件开发过程中，复杂的功能系统可以分解为对象类单元集合，对象类之间的依赖和交互可以实现软件系统的各项功能。

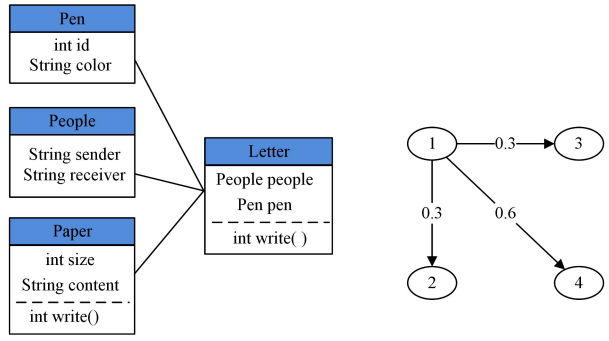
在图 1(a)中，对象类 Letter 继承了对象类 Paper，并且使用对象类 People 和 Pen 作为 Letter 的成员，这样的关系可以在类图(见图 1(b))中显示出来。而在软件网络中，对象类之间的关系由权值决定，因此需要对对象类之间的依赖关系进行度量，并在软件网络中进行加权处理：

$$W_{i\to j}=(\alpha\times R_{i\to j}+\beta\times CK_{i\to j}) \tag{1}$$

其中，$\alpha$ 和 $\beta$ 分别是对象类关系强弱和 CK 度量的加权系数，并且满足 $\alpha+\beta=1$。

```
class Pen{
    int id;
    String color;
}
class Paper{
    int size;
    String content;
    int write()
}
class People{
    String sender;
    String receiver;}
class Letter;public Paper{
    People people;
    Pen pen;
    int write();
}
```

(a)源代码




(b)类图　　(c)软件网络

图 1　代码、UML 类图和软件网络的关系

Fig. 1　Relationship of code, UML and software network

1)UML 中类之间的五大关系

在面向对象软件网络中，边表示两个对象类节点之间的直接关系，其中对边进行加权可以更好地表示对象类节点之间的调用关系和重要性等。这种调用关系可以反映出，在软件工程类的设计中有相似功能特征的对象类会放在同一个包或者模块中。在 UML 中，类之间的关系主要有依赖、关联、聚合、组合以及泛化。

依赖关系表示一个对象类要完成某个功能需要引用另一个对象类，其中某个对象类的变化将部分影响另一个对象类。关联关系是指两个对象类之间存在连接关系，这种连接关系比依赖关系更强。具有依赖关系的两个对象类都不会增加对象的属性，只是单纯地利用关系，而发生关联的两个对象类中，其中一个对象类或者部分属性会成为另一个对象类的属性，使得这两个对象类更加耦合。聚合关系属于关联关系，偏

向于某个对象类包含另一个对象类的关系。组合关系是一种多聚合关系，表示一个对象类依赖于多个对象类，并且包含这些对象类。泛化关系表示一般元素和特殊元素之间的分类关系，在实际开发过程中，泛化关系是一种对象类继承关系、接口和实现关系，因此对象类之间的强弱关系 $R_{i\to j}$ 如表 1 所列(D1－D5 代表对象类之间的关系逐渐变强)。

表 1 对象类之间的关系

Table 1 Relationship of object classes

| 关系 | 关系强弱 |
|---|---|
| 依赖(dependence) | D1 |
| 关联(association) | D2 |
| 聚合(aggregation) | D3 |
| 组合(composition) | D4 |
| 泛化(generalization) | D5 |

2)对象类度量集

传统的软件度量主要针对以功能分解和数据流分析为核心的结构化软件开发范型。由于面向对象软件开发范型与传统结构化软件开发范型有显著区别，因此传统软件度量并不适用于面向对象的开发过程。目前，面向对象软件度量的理论研究和实践已成为软件工程领域的研究热点之一，在面向对象软件网络中，使用 CK 度量组[14]中的改进度量 WMC(每类加权方法数，记为 $W_i$)以及 LCOM4(方法缺乏类聚数，记为 $L_i$)对软件网络中的对象类关系进行加权。这是因为在系统开发过程中，当功能增加，对象类的方法也随着增加时，对象类凝聚力趋向于变弱，有些模块被添加到对象类中。

$$CK_{i\to j}=(W_i\times\alpha)+(L_i\times\beta) \tag{2}$$

其中，$\alpha$ 和 $\beta$ 分别是 WMC 和 LCOM4 的加权系数，并且满足 $\alpha+\beta=1$。

### 2.2 分形维数的计算

对于软件网络，可以使用紧凑盒子燃烧(Compact Box Burning，CBB)算法来计算其分形维度。但在 CBB 算法中，网络中的节点通过随机选择的方式进入盒子，这种盒子节点选择方式具有随机性，导致其破坏了整个网络的结构；尤其是在软件网络中，网络中节点的具有复杂的依赖关系，需要增加盒子数来覆盖整个软件网络，导致网络中的盒子覆盖节点准确率降低，且不能保证软件网络中节点的连通性。

针对以上方法的缺陷，文中采用基于网络中心性的盒子计数法来计算软件网络结构中的分形维数。具体步骤如下：

1)确定盒子尺寸 $l_b$，构造集合 $C$，该集合包含所有未覆盖节点。

2)选取网络中心性最大的节点 $i$，选取距离节点 $i$ 的最短路径 $l_i$ 小于 $l_b$ 的节点作为盒子内的节点。

3)统计盒子尺寸 $l_b$ 下能完全覆盖整个网络的盒子数 $N_b(l_b)$。

4)重复步骤 2)和步骤 3)，直到未覆盖节点集合 $C$ 为空，计算出分形维数 $d_b$ 作为交互网络的分形结构特征，具体如下：

$$d_b\approx\frac{\ln(N_b(l_b))}{\ln(l_b)} \tag{3}$$

其中，$d_b$ 是盒子 $b$ 的分形维数，$l_b$ 是盒子 $b$ 的尺寸，$N_b(l_b)$是盒子 $b$ 覆盖整个网络的最小盒子数。

## 3 实验结果与分析

本节主要利用盒子计数法对 spring 和 struts2 软件网络进行测算，3.1 节介绍了数据采集方法，3.2 节对软件网络及其模块的软件网络进行了分形特征的测算以及分析，3.3 节对版本演化的软件网络进行了分形特征的测算以及分析。

### 3.1 数据采集

本文对目前 Java web 开发最流行的两个开源框架 spring 和 struts2 进行数据采集，并利用 Class Dependency Analyzer 2.2.0 软件抽取这两个框架重用模块中对象类之间的依赖关系，构建网络软件来分析模块间的分形特征。除此之外，采集了不同版本的 spring 框架中的重要模块软件网络，以探究版本间的演化关系。

### 3.2 模块分形特征测算结果

利用基于网络中心性的盒子计数法对 spring 和 struts2 软件网络进行分析维数的测算。文中对 spring 以及 spring 的 aop，aspect，beans，tx 和 web 模块进行了分析，结果如图 2 所示；并对 struts2 以及 struts2 的 core，dojo，emb，gxp 和 jsf 模块进行了分析，结果如图 3 所示。结果显示，spring，struts2 及其主要模块软件网络中，其盒子数和盒子大小呈对数线性关系，具有基本分形性质。

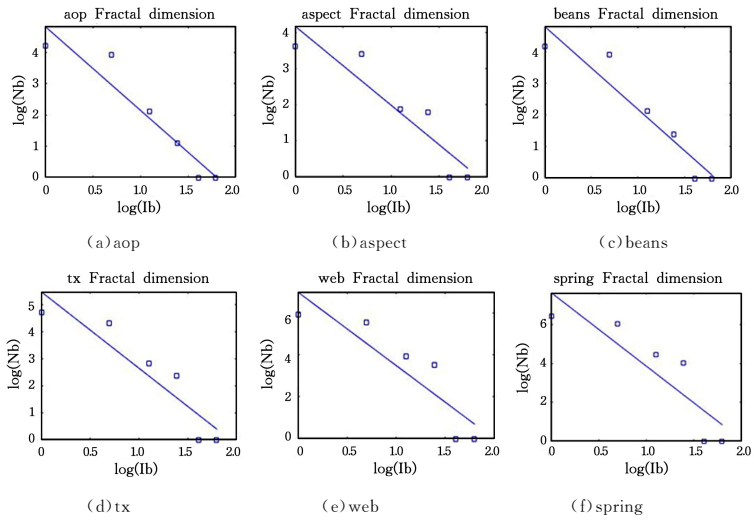



图 2 spring 框架及其主要模块的分形特征

Fig. 2 Fractal of spring framework and its modules

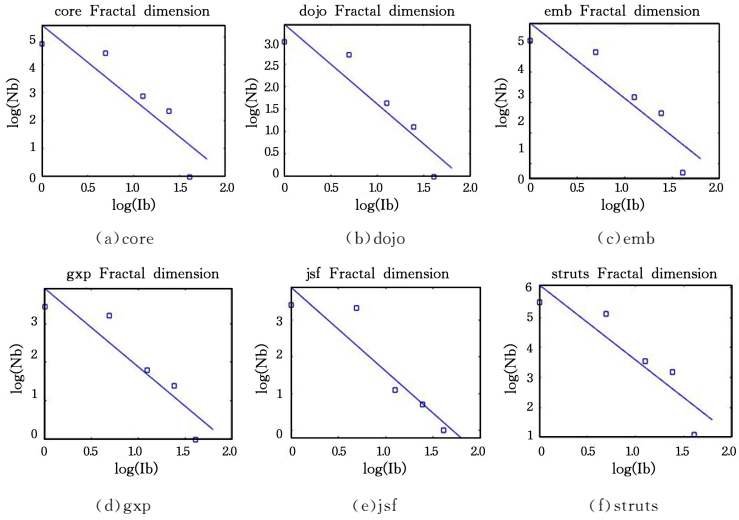



图 3　struts2 框架及其主要模块的分形特征

Fig. 3　Fractal of struts2 framework and its modules

在利用盒子计数法计算软件网络分形结构特征时，对象类之间的调用深度可以由盒子大小体现。假设某个盒子长度中整个网络需要更多的盒子来进行覆盖，则该软件网络中对象类间的调用关系更复杂，对象类对整个模块的功能覆盖程度越高，其功能愈加完善，分形维数越大。因此，可以用分析维数定义对象类软件网络的覆盖程度，在 spring 和 struts2 框架中，模块的分形维数变化范围约为 2～3.7，不同模块的功能相差较大，其分形维数变化也具有一定的差异性。具体分形维数如表 2 和表 3 所列。

表 2　spring 主要模块的分形维数

Table 2　Fractal dimension of spring modules

| 模块名称 | 分形维数 | 模块功能 |
|---|---|---|
| aspect | 2.1902 | 基础功能模块 |
| beans | 2.6080 | |
| aop | 2.6690 | |
| tx | 2.8255 | |
| core | 3.0696 | 公用核心模块 |
| context | 3.1492 | |
| jdbc | 3.2146 | 综合应用模块 |
| web | 3.5065 | |
| mvc | 3.6383 | |

表 3　struts2 主要模块的分形维数

Table 3　Fractal dimension of struts2 modules

| 模块名称 | 分形维数 | 模块功能 |
|---|---|---|
| dojo | 1.8473 | 基础功能模块 |
| jsf | 2.1219 | |
| gxp | 2.2349 | |
| core | 2.8749 | 核心应用模块 |
| emb | 2.8553 | |

模块的作用决定了其功能的综合性和复杂性，由于模块的功能不同，其对象类的覆盖程度也不相同。以 spring 框架为例，在实际软件开发中，spring 多用于大型 java web 系统的开发，利用 spring 和 spring mvc 框架能够有效地提高开发效率。因此，spring 模块中，mvc 和 web 模块封装了大量的 servlet 方法用于 web 系统的开发。而大型 web 系统需要与数据库进行交互，java 系统通常需要通过 jdbc api 对进行数据库操作。tx，aop，aspect 和 beans 属于功能性模块，为软件系统结构提供中介功能。例如，jdbc 模块中需要 tx 模块对数据库进行事务处理，aop 模块需要结合 aspect 模块为 web 和 mvc 模块实现面向切面的操作，为其运行提供生命周期。context 和 core 作为 spring 框架的核心模块，为其他模块提供一些常用的公用方法。因此，按照实际开发过程中功能的复杂程度对 spring 中各个模块进行排序应当是 mvc，web，jdbc＞context，core＞tx，aop，aspect，beans。在表 2 和表 3 中，综合性较高的模块需要更多的对象类对该模块进行覆盖，因此平均分形维数要大于公用类模块和功能性模块。

### 3.3　版本演化分形特征测算结果

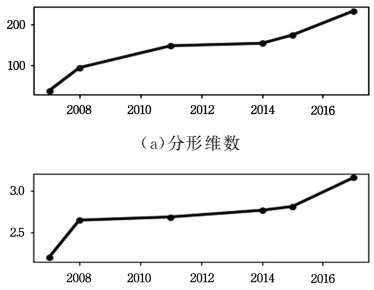
利用基于网络中心性的盒子计数法对 spring 中的核心模块 context 不同版本的软件网络进行分形维数的测算，得到如表 4 所列的结果。结果显示分形维度随着版本的更新也在增加，从 2007 年开始，java web 越来越成为 web 系统应用的主流开发框架，由于 spring 一直是各大框架的粘稠剂，使用率逐年增长（见图 4），因此为了完善 spring 框架的功能，为开发者提供更方便、迅捷的开发模式，spring 中各个模块随着版本的迭代，其功能也越来越完善，对象类之间的依赖关系也愈加复杂，软件网络中对象之间的覆盖率也越来越高。

表 4　spring context 版本演化分形维数

Table 4　Fractal dimension of spring context version evolution

| 模块名称 | 分形维数 | 使用率 | 时间 |
|---|---|---|---|
| 2.0 | 2.2076 | 37 | 2007 |
| 2.5 | 2.6494 | 93.75 | 2008 |
| 3.0 | 2.6874 | 147.5 | 2011 |
| 3.2 | 2.7662 | 153.6 | 2014 |
| 4.1 | 2.8104 | 173.9 | 2015 |
| 4.3 | 3.1492 | 233 | 2017 |

实际上，spring 框架在版本的迭代过程中并不需要开发者对现有软件系统结构进行更改，并且还能保证功能的增加与完善。由图 4 可以看出，spring context 模块、使用率和分形维数随着版本的迭代而增长，并且两者的增长趋势类似。




(a)分形维数

(b)使用率

图 4 spring context 分形维数和使用率

Fig. 4 Fractal dimension and usage coefficient of spring context

在开源社区中,spring 的使用率一直保持上涨,由于其功能强大且更新稳定,一直受到大型项目的青睐。使用率的增加一方面代表着开发者对新增功能满意程度的增加;另一方面,新功能的增加并没有破坏原有软件系统的结构,在保证原有软件系统结构的情况下对系统功能进行完善。

**结束语** 文中通过采集软件系统中对象类的依赖关系构建软件网络,并提出了一种基于网络中心性的盒子计数法对软件网络的分形特征进行测算。结果显示,spring 和 struts2 两款主流 web 系统开发框架的软件网络具有分形特征,总体上体现了软件结构中对象类的依赖关系具有自相似的特征。然而各个模块之间的分形维数在数值上存在明显的差异,这样的差异也体现了模块的功能差异:在软件系统架构中,综合性应用模块功能较为复杂,其涉及到的对象类较多,依赖关系复杂并且网络覆盖率高;而基础功能模块封装性较强,功能比较专一,依赖关系也比较专一,网络覆盖率低。

随着版本的演化,软件系统规模也在扩大,其依赖关系和对象类网络覆盖率也在增加。理论上来说,在框架开发过程中,随着使用率的增加,框架的更新速度加快,其功能也越来越完善,网络的覆盖率也会提升。尽管框架的使用率变化趋势和分形维数有类似的表现,但除系统结构的合理性影响使用率之外,功能的完善还与开源社区以及公司的选择密切相关,两者的关系还需要进一步讨论。

对象类之间的依赖关系与系统的设计息息相关,度量其复杂性还需要进一步的研究,使得今后对软件网络分形结构特征的研究更加符合软件系统架构状况。通过从对象类的依赖关系角度研究软件网络的结构特征,为软件工程中的需求、开发、测试等阶段提供了一种新的研究思路,是一个有意义的研究课题。

## 参 考 文 献

[1] WILFREDO T. Software Fault Tolerance: A Tutorial[J]. Pomales, 2000, 1(2): 220-232.

[2] BRIAND L C, DALY J W, WÜST J. A Unified Framework for Cohesion Measurement in Object-Oriented Systems[J]. IEEE Transactions on Software Engineering, 2002, 25(1): 91-121.

[3] MYERS C R. Software systems as complex networks: structure, function, and evolvability of software collaboration graphs[J]. Physical Review E Statistical Nonlinear & Soft Matter Physics, 2003, 68(2): 046116.

[4] DONG Y, QI G N. The scale-free feature and evolving model of largescale software systems[J]. Acta Physica Sinica, 2006, 55(8): 3799-3804. (in Chinese)
闫栋,祁国宁. 大规模软件系统的无标度特性与演化模型[J]. 物理学报,2006,55(8):3799-3804.

[5] HAN M C, LI D Y, LIU C Y. Networked characteristics in software and its contribution to software quality[J]. Computer Engineering & Applications, 2006, 42(20): 29-32.

[6] MA Y T, HE K Q, LI B, et al. Empirical Study on the Characteristics of Complex Networks in Networked Software[J]. Journal of Software, 2011, 22(3): 381-407.

[7] GAO Y, PENG Y, XIE F, et al. Software Metrics for complex networks[J]. Journal of Tsinghua University(Science and Technology), 2012, 52(10): 1474-1481. (in Chinese)
高洋,彭勇,谢丰,等. 基于复杂网络的软件结构度量[J]. 清华大学学报(自然科学版),2012,52(10):1474-1481.

[8] CHONG C Y, LEE S P. Analyzing maintainability and reliability of object-oriented software using weighted complex network [J]. Journal of Systems & Software, 2015, 110(C): 28-53.

[9] WANG B Y, LV J H. Software Networks Nodes Impact Analysis of Complext Software Systems[J]. Journal of Software, 2013, 24(12): 2814-2829. (in Chinese)
汪北阳,吕金虎. 复杂软件系统的软件网络结点影响分析[J]. 软件学报,2013,24(12):2814-2829.

[10] KUANG L, ZHENG B J, LI D Y, et al. A fractal and scale-free model of complex networks with hub attraction behaviors[J]. Science China Information Sciences, 2015, 58(1): 1-10.

[11] CONCAS G, LOCCI M F, MARCHESI M, et al. Fractal dimension in software networks [J]. Europhysics Letters, 2006, 76(6): 1221-1227.

[12] CONCAS G, MARCHESI M, MURGIA A, et al. Assessing traditional and new metrics for object-oriented systems[C]// Proceedings of the 2010 ICSE Workshop on Emerging Trends in Software Metrics. ACM, 2010: 24-31.

[13] TURNU I, CONCAS G, MARCHESI M, et al. The fractal dimension of software networks as a global quality metric[J]. Information Sciences, 2013, 245(10): 290-303.

[14] CHIDAMBER S R, KEMERER C F. A metrics suite for object oriented design[J]. IEEE Transactions on Software Engineering, 1994, 20(11): 197-211.