面向软件执行网络的行为拓扑分析研究

张锡哲 罗 实 印 莹 张 斌

(东北大学信息科学与工程学院计算机应用技术研究所 沈阳 110004)

摘 要 随着互联网及软件技术的逐渐成熟和发展,软件系统将面临使用模式不确定、动态交互行为复杂变化的问题。针对软件执行过程中的交互复杂性,分析了基于复杂网络的软件执行网络行为拓扑度量特征。以Linux下3个典型的开源软件为研究对象,首先获取其软件执行交互记录,并根据方法调用关系构建执行网络,然后分析软件执行网络的连通性、网络直径与密度、平均路径长度、度分布、度相关性、聚集系数、介数、接近度等典型拓扑度量,并与以代码静态关联为基础构建的软件结构网络进行对比分析。结果表明,软件执行过程中具有执行行为动态变化和行为重组现象,其拓扑特征与结构网络具有较大差别,执行网络平均路径长度变小,其小世界特性趋于消失。探索软件执行行为规律对于软件运行维护及质量保障具有重要的指导意义。

关键词 复杂网络,开源软件,执行行为,方法调用,拓扑特征

Analysis on Dynamic Behavior for Open-source Software Execution Network

ZHANG Xi-zhe LUO Shi YIN Ying ZHANG Bin

(Information Science and Engineering, Northeastern University, Shenyang 110004, China)

Abstract With the development and increasingly mature of Internet and software technology, software system is facing with the characteristic of uncertain using pattern and complex dynamic interaction behavior. This paper analysised the topological measure characteristics in software execution network based on the complex network. Take three typical open-source software running in Linux as an example, first the software execution log and according to the calls relationship of methods the execution network were established, and then typical topological metrics like connectivity, network diameter and density, average path length, degree distributions, degree correlations, clustering coefficient, betweenness centrality and closeness were analyzed in the software execution network, also make comparison with software structural network based on static code association. The results show that execution behavior have dynamic changes and restructuring phenomenon in the software execution process, compared with the structural network the topological characteristics in the execution network have many difference, the average path length becomes smaller in the perform network, the small-world character tends to disappear. Explore software execution behavior rules has important guiding significance for the software maintenance management and quality assurance.

Keywords Complex network, Open-source software, Dynamic execution, Function call, Structural characteristic

1 引言

随着互联网计算技术(如SOA, SaaS[1]等)的快速发展,软件产品呈现出更多的模块化、网络化特征。服务封装、动态重构及多租户等技术的应用,已使得软件系统的复杂性从创建时的结构复杂逐步转变为执行时行为复杂。研究并理解软件系统的执行时行为特征已成为当前研究的热点问题。

软件作为一类人造复杂系统,具有结构复杂、难操控的特点,这给软件的设计、开发、维护和管理等造成了诸多困难。近年来,复杂网络[2-4]理论的成熟及其带来的新发现为理解分析复杂系统提供了强有力的手段。通过将软件系统看作复杂网络,从整体和全局的角度来探索和发现软件的结构特性、动态行为和演化规律,有助于全面地理解软件系统的本质特征及量化软件的复杂性,在提高软件效率和保障软件质量等方面具有重要意义[5]。

基于复杂网络的软件网络的构建方法主要有基于软件元素间关联关系的结构网络[6-10]和基于软件元素间执行调用关系的执行网络[11,12]两类。结构网络侧重于软件的静态拓扑结构的度量,通过抽取软件系统的不同实体粒度(如方法、对象、类、包、构件等)以及它们之间的联系构建网络,大多发现了复杂网络中存在的"小世界"和"无标度"特性[6,8],并猜测这可能与软件设计方法学中所提倡的"高类聚、低耦合"和软件工程中的模块重用原则有关[7]。而执行网络针对软件执行时的方法调用关系构建网络,关注软件的动态执行过程,以一

本文受国家自然科学基金(60903009,61073062),中央高校基本科研业务费专项资金(90104001)资助。

张锡哲(1978—),男,博士,副教授,主要研究方向为复杂网络、服务计算,E-mail:zhangxizhe@ise.neu.edu.cn;罗 实(1986—),男,硕士生,主要研究方向为软件网络;印 莹(1980—),女,博士,讲师,主要研究方向为服务计算;张 斌(1964—),男,博士,教授,博士生导师,主要研究方向为服务计算。

种进化的观点审视软件系统，从而揭示出软件系统在运行时所存在的一些潜在规律或满足的一些不变模式。这类方法抽取了大量软件运行时的对象快照，以对象为节点、对象间的引用为边，发现了对象引用网络中也存在幂率分布。

下一代的网络化软件结构特征趋于消失，系统复杂性体现在运行时动态重组和执行行为以及不确定的用户需求上。软件系统很多特性是在运行过程中体现的，静态研究只是从系统结构层面进行了统计分析，不能帮助人们充分理解系统的内在行为特征。例如，GUI 软件系统内部存在大量的分支、循环等控制结构，它随用户的不同输入呈现不同的状态；又如在面向对象的软件系统中，由于多态性和动态绑定机制的存在，系统的行为特征只有在运行期间才能被真正确定。所以，以软件执行网络为研究对象分析软件执行行为具有重要的意义。

本文以 Linux 下 3 个典型开源软件为研究对象，将其在实际运行环境中得到的方法调用序列看作一类软件执行网络，给出了相应的网络模型并定义了平均路径长度、聚类系数、度分布等网络结构特征，并与静态拓扑结构下的拓扑特征进行了对比和分析，分析其形成原因及特征。相关结论揭示了软件系统的动态执行特性，对于软件运行维护及质量保障具有重要的指导意义。

本文第 2 节给出了软件执行网络的模型和结构特征的定义；第 3 节介绍了实验方法和环境；第 4 节为实验结果和分析；最后进行了总结。

2 软件网络模型及度量

软件执行网络是软件系统在执行过程中函数之间的相互调用所形成的网络，它以函数实体为网络节点，把函数调用 $i \to j$ 看作是节点 i 到节点 j 的一条有向边。如图 1 所示，左边部分为一段执行时的方法调用序列，右边部分为与其对应的软件执行网络拓扑图。执行网络 G 可用三元组 $\langle V, E, A\rangle$ 进行描述，其中 $V=\{v_1, v_2, \cdots, v_m\}$ 是网络中节点的集合，$E=\{e_{ij}; i,j=1,2,\cdots,m\}$ 是网络中边的集合，A 是网络的属性集合，分为节点属性和边属性，具体定义如表 1 所列。

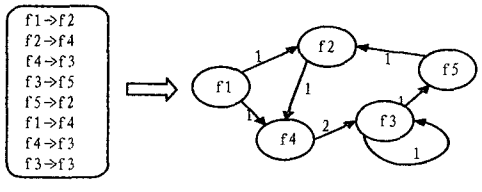


图 1　函数调用序列所对应的执行网络

表 1　网络属性定义

属性表示	属性名称	意义
nid	节点序号	唯一标识每个节点的编号
name	节点名	节点对应的函数名称
cost	节点执行代价	函数自身所消耗的执行时间
k_{in}	节点入度	函数被多少其它函数所调用
k_{out}	节点出度	函数调用了多少其它函数
s_{in}	入点强度	函数被调用的总次数
s_{out}	出点强度	函数调用其它函数的总次数
eid	边序号	唯一标识每条边的编号
source	源点	边的起始节点的序号
target	终点	边的终止节点的序号
weight	权重	单对函数调用发生的次数

在执行网络中，节点强度是对拓扑结构中节点度的扩展，它既考虑了节点的邻居数，又考虑了该节点与邻居节点的权重。值得注意的是，节点执行代价并不是函数单次自身调用所消耗的时间，而是整个执行过程中函数总的调用所消耗的时间，这跟节点的强度有一定关系。

下面给出网络的基本度量指标：

a. 连通组件(Connected Component)

若图 G 中存在一个子图 G_1，G_1 中所有节点都相互可达，则 G_1 就是 G 的一个连通组件。在无向图中的连通组件称为弱连通组件(WCC)，有向图中的连通组件称为强连通组(SCC)。当然，在有向图中同时存在以上两种连通组件。

b. 网络直径(Network Diameter)

网络中两个节点 i 和 j 之间的距离 d_{ij} 定义为连接这两个节点的最短路径上的边数，网络中任意两个节点之间距离的最大值称为网络的直径。

c. 网络密度(Network Density)

网络密度为网络中实际存在的边数 M 比上理论上最多的边数，无向网络和有向网络分别为：

$$Density_{undirected}=\frac{M}{\frac{1}{2N(N-1)}}$$

$$Density_{directed}=\frac{M}{N(N-1)}$$

d. 平均路径长度(Average Path Length)

网络的平均路径长度 APL 表示任意两个节点之间距离的平均值，无向网络和有向网络分别为：

$$APL_{undirected}=\frac{1}{\frac{1}{2N(N-1)}}\sum_{i\neq j}d_{ij}$$

$$APL_{directed}=\frac{1}{N(N-1)}\sum_{i\neq j}d_{ij}$$

e. 聚类系数(Clustering Coefficient)

假设一个无向网络中节点 i 有 k_i 条边将其与其它节点相连，则节点 i 就有 k_i 个邻居，这 k_i 个邻居最多存在 $k_i(k_i-1)/2$ 条边，那么节点 i 的聚类系数就是这 k_i 个节点实际存在的边数 m_i 比上可能最多存在的边数 $k_i(k_i-1)/2$，即 $CC_i=2m_i/(k_i(k_i-1))$。在有向网络中节点 i 的聚类系数为：$CC_i=m_i/(k_i(k_i-1))$，而整个网络的聚类系数就是网络中所有节点聚类系数的平均值：$CC=\frac{1}{N}\sum_{i}^{N}CC_i$。

f. 度分布(Degree Distribution)

节点 i 的度定义为与该节点相连的其它节点的数目，在有向网络中节点的度可分为入度和出度。网络中所有节点的度的平均值为网络的平均度 $\langle k\rangle$。度分布用 $p(k)$ 描述，它表示一个随机选定的节点的度恰好是 k 的概率。累积(Cumulative)度分布 $P(k)$ 同样可以用来描述度的分布情况，它表示度不小于 k 的节点的概率。若度分布 $p(k)$ 满足以下幂律函数，则符合“无标度”分布。

$$p(k)\sim k^{-\lambda}$$

那么累积度分布 $P(k)$ 则满足：

$$P(k)=\sum_{k'=k}^{\infty}p(k')\sim k^{-(\lambda-1)}$$

g. 度相关性(Degree Correlation)

网络中的每个节点的度相关性用其入度和出度的数值对 (k_{in}, k_{out}) 表示，网络中的节点入度集 $\{k_{in}\}$ 和出度集 $\{k_{out}\}$ 的相关性可用皮尔逊(pearson)线性相关系数表示，它描述了网络

入度和出度这两个定矩变量联系的紧密程度，有：

$$DC=\frac{\sum(\{k_{in}\}-\langle k_{in}\rangle)(\{k_{out}\}-\langle k_{out}\rangle)}{\sqrt{\sum(\{k_{in}\}-\langle k_{in}\rangle)^2\sum(\{k_{out}\}-\langle k_{out}\rangle)^2}}$$

h. 介数(Betweenness)

假设节点 i 和 j 之间存在的最短路径数目为 g_{ij}，点 i 和 j 之间存在的经过节点 k 的最短路径数目为 $g_{ij}(k)$，那么节点 k 处于节点 i 和 j 的最短路径数目的概率 $b_{ij}(k)=g_{ij}(k)/g_{ij}$，它描述了节点 k 对于 i 和 j 的交往控制能力。那么节点 k 的介数值为：

$$Betweenness_k=\sum_{i}^{N}\sum_{j}^{N}b_{ij}(k),i\neq k\neq j$$

节点的介数反映了其在整个网络中的作用和影响力，具有很强的现实意义。

3 研究方法与对象

3.1 软件执行网络构建方法

为了能在运行开源软件期间对方法调用关系进行追踪，本文使用 Gnu 编译工具链对开源软件的源程序进行调试编译，使其能在执行时生成必要的调试信息文件 gmon. out，然后利用 Gnu 环境下的 Gprof 工具对 gmon. out 进行分析。Gnu/Gprof 是类 Unix 平台下对 C/C++开源项目的一个 profile 分析工具，它能在程序运行过程中记录下函数间的调用关系、每个函数被调用的次数、每个函数消耗的时间等代码级信息。gcc 编译器在程序的每个函数中加入一个名为“mcout”(或“_mcount”，依赖于编译器或操作系统)的函数，该函数在内存中保存了一张函数调用图，可利用函数调用堆栈的形式查找子函数和父函数的地址，从而获得函数间的调用关系，以及每个函数调用次数、运行时间等信息。Gprof 是一个源码分析工具，它能打印出每个方法调用的次数、方法之间的调用关系以及方法调用消耗时间等运行时信息。为了从这些运行时信息中获得需要的方法调用关系，我们利用特定的脚本文件对 Gprof 的分析结果进行信息过滤，抽取方法调用关系形成序列文件，最后根据网络模型对函数调用序列进行网络结构特征计算，得到结果。

另外，为了模拟软件在真实环境中被用户使用的情况，实验中对每个开源软件生成了一个用例库，每个用例库包含 100 个用例，每个用例都是使用该软件的一个实际案例，它描述了用户与软件的交互行为，由一系列连续的按钮动作组成。实验环境如图 2 所示。

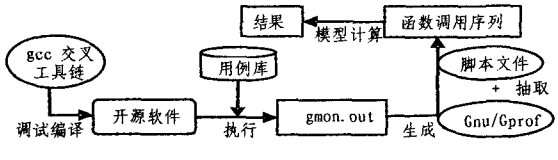


图 2 实验流程

3.2 实验对象

本文以 Linux 环境下典型的开源 GUI 软件作为对象，在真实使用的环境下对其生成的执行网络的结构特征以及特征的动态演化情况进行研究。选取的开源 GUI 软件如下。

1. Gimp 是 Gnu/Linux 平台下的一款图像处理软件，类似于 Windows 下的 Photoshop。它包含了大量功能模块来提供对图像处理的众多要求，同时也允许添加额外的插件来实现特殊的功能。其界面接口如图 3 所示。

图 3 Gimp 的界面接口

2. Dia 是一款基于 GTK+的图形绘制工具，它内置了大量的图形对象以实现流程图、UML 图、网络图、电路图等多种图形的绘制工作；此外用户还可以通过编写 XML 文件添加新的图形形状来满足特殊的绘制要求，其界面接口如图 4 所示。

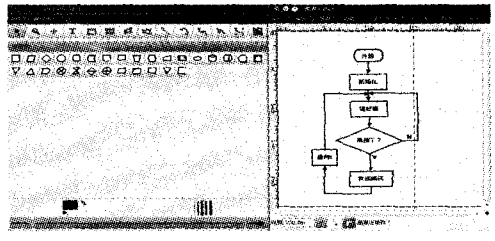


图 4 Dia 的界面接口

3. Amule 是一款基于 XMule 和 IMule 的跨平台 P2P 文件共享软件，类似于 eMule，可同时支持 EDonkey 和 Kad 网络，其界面接口如图 5 所示。

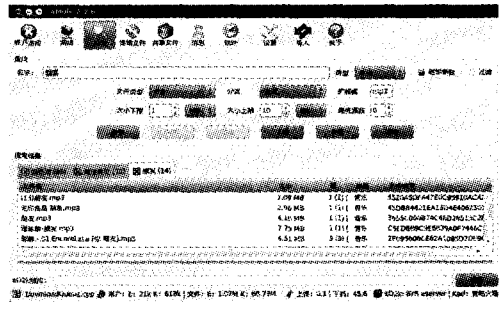


图 5 Amule 的界面接口

以上 3 款都是基于 GTK+界面风格的开源软件，遵循 GNU 通用公共许可证协议，可以免费地获得和使用。它们操作简单、功能丰富实用，已成为各自应用领域内受广泛欢迎的软件产品，具有一定的代表性和研究价值。

4 结果与分析

针对单个开源软件，每次从其用例库中随机抽取一个用例执行后生成函数调用序列文件，通过序列文件构造网络，图 6 所示的是 Gimp 单个用例所形成的执行网络。

按照复杂网络结构参数定义计算执行网络的各指标值，随机取各软件的单个用例进行实验，重复 20 次后取平均值。统计结果如表 2 所列。

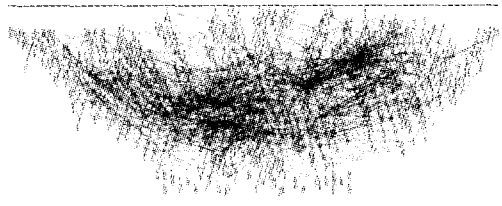


图 6 Gimp 单个用例所生成的执行网络

表 2　执行网络结构特征统计信息

	Amule	Dia	Gimp
[N,M]	[9645.1,17640.0]	[539.5,957.15]	[3842.5,10190.8]
$[N,M]_{WCC_{max}}$	[9542.4,17573.75]	[526.8,947.6]	[3642.15,10077.4]
Diameter	20.15	9.5	34.95
Density	1.748E−4	3.180E−3	1.067E−3
APL	5.205	2.794	9.404
CC	0.018	0.035	0.071
DC	0.016	−0.128	−0.034
$\langle k_{in} \rangle$	1.784	2.352	3.607
$\langle k_{out} \rangle$	1.707	2.883	3.239

另外通过分析各开源软件中所有源代码的函数依赖关系，得到静态网络的结构统计指标如表 3 所列。

表 3　静态网络结构特征统计信息

	Amule	Dia	Gimp
[N,M]	[19368,36993]	[4854,17772]	[17591,97403]
$[N,M]_{WCC_{max}}$	[10040,28813]	[4655,17433]	[17317,97195]
Diameter	26	14	19
Density	9.862E−5	7.544E−4	3.148E−4
APL	6.218	3.479	4.024
CC	0.011	0.030	0.042
DC	0.028	−0.077	−0.030
$\langle k_{in} \rangle$	2.347	4.907	6.622
$\langle k_{out} \rangle$	4.752	6.070	9.148

4.1　连通组件

3 个软件执行网络中包含的弱连通组件数平均情况下分别为 41.3,5.7,87.25。在这些弱连通子图中，有一个规模相比其它大得多的组件(表 2 中用 WCC_{max} 表示)，它所含节点数占整个网络节点的比例分别为 98.94%,97.65%,94.79%。在静态网络中的弱连通子图数分别为 1185,33,72，其最大弱连通组件节点所占的比例分别为 51.84%,95.90%,98.44%。对于强连通性，无论是执行网络还是静态网络几乎都不存在节点数大于 1 的组件，说明网络中节点的连接具有很单一的方向性。

结合软件系统的本质很容易理解网络中缺少大规模强连通子图的现象。对于函数调用关系来讲，若函数 a 调用了函数 b 来共同完成某项功能，那么函数 b 就成为函数 a 实现的一部分，反过来通常是不成立的。同样，对于类协作网络中的继承关系，子类到父类是可达的，因为父类封装了最基本的定义以便子类重用，所以如果父类到子类也可达就会破坏面向对象设计中的封装、继承思想，也不符合软件工程的重用原则。

4.2　网络直径与密度

3 个软件平均情况下执行网络和静态网络的直径分别是 20.15,9.5,34.95 和 26,14,19；密度分别是 1.748E−4,3.180E−3,1.067E−3 和 9.862E−5,7.544E−4,3.148E−4。其中 Amule 和 Dia 的执行网络直径比静态网络要小，表明在软件执行过程中没有覆盖最长距离的函数调用路径；而 Gimp 却相反，网络中最长的调用路径变大了，出现此现象的原因是编译器在编译源程序期间生成了新的函数，该函数在执行过程中动态地对调用路径进行了重组，结果导致网络的直径增大，这也说明执行网络并不是简单地选择覆盖静态网络中的函数关联路径，而是包含了一些节点与边的增加、路径重组等复杂操作。另外，3 个软件的网络密度在静态和动态下都很小，表明函数网络是一类稀疏网络，网络中实际存在的边数远没达到全局耦合网络中理论存在的最大值，但动态下网络密度要稍大，这是由于执行网络按需将能协作完成某些特定功能的节点动态绑在了一起，将另一些不相关的节点进行了隔离，因此呈现出一种相对“密集”的效果。

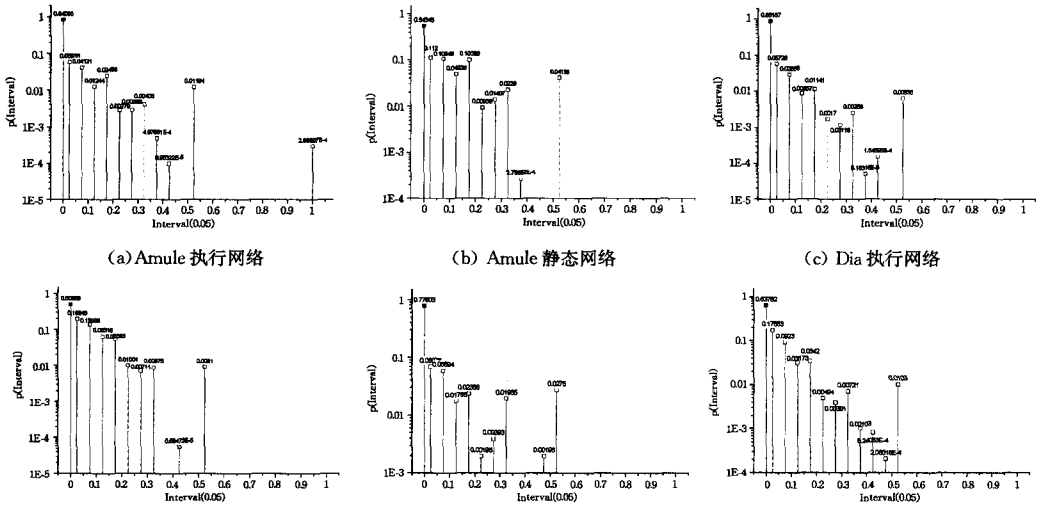
4.3　平均路径长度

3 个软件在平均情况下的执行网络和静态网络中拥有的最短路径数目分别是 276512.7,3998.35,149243.8 和 1853511,99529,806540。平均路径长度分别为 5.205,2.794,9.404 和 6.218,3.479,4.024，相比网络规模其值都很小，表明函数网络在动态和静态下都具有“小世界”的特性。

类似于网络直径，Amule 和 Dia 在动态下的平均路径长度变小了，而 Gimp 却变大了。变小是由于这两个软件的执行网络中所拥有的最短路径数目大量减少，并保留下来的都是一些较短长度的路径造成的。而对于变大的原因，除在网络直径中已分析的因素外，Gimp 系统本身高度模块化和用例功能的单一性也可能造成平均路径长度变大，因为单一功能的用例在执行中所涉及的程序流程不会很复杂，导致大量模块间出现的短路径减少，增加了函数间可达路径的长度。

4.4　聚类系数

对于聚类系数，无论是执行网络中的 0.018,0.035,0.071，还是静态网络中的 0.011,0.030,0.042，都远远大于它们在同等规模下随机网络的 $O(N^{-1})$。这说明函数网络是一类高聚类网络，网络中的函数具有成团簇拥的聚集现象。



(a) Amule 执行网络　(b) Amule 静态网络　(c) Dia 执行网络

(d) Dia 静态网络　(e) Gimp 执行网络　(f) Gimp 静态网络

图 7　执行网络和静态网络的聚类系数分布

伴随着网络密度的增加，聚类系数在动态下变大很容易理解。图 7 给出 3 个软件单个用例的执行网络和静态网络中每个节点的聚类系数分布情况。其中横坐标表示聚类系数值的区间(分为(0,0.05),[0.05,0.1),[0.1,0.15),…,[0.9,0.95),[0.95,1)和两个特殊的区间[0,0],[1,1])，纵坐标表示聚类系数落在对应区间的节点概率。

从图 7 可以看出，无论是动态网络还是静态网络，聚类系数为 0 的节点在网络中所占的比例都最高，且都超过了 50%。对于 Dia 和 Gimp，网络中节点的最大聚类系数为 0.5，只有理论最大值 1 的一半，进一步说明了软件函数网络具有很强的方向单一性。

4.5 度分布

在平均情况下，Dia 执行网络的入度平均值均小于出度平均值，这是因为网络中存在有入度的节点数目大于有出度的节点数目，说明了软件在执行过程中有更多的函数倾向于被调用而不是调用其它函数。相反，Amule 和 Gimp 执行网络中似乎有更多的函数需要其它函数的协作才能正常工作，这导致了它们的入度平均值大于出度平均值，同样的情况在 3 个软件的静态网络中都得到了体现。另外，通过对比发现，网络中的入度和出度平均值差异从静态网络中的最低 25.21%(Dia)下降到动态执行网络中的最高 22.58%(Dia)，标志着网络在执行过程中有入度和出度的节点比例分配趋于平衡，可能对动态系统的稳定性具有一定的作用。图 8 是各软件单个用例的入度和出度散点分布情况，其中横坐标表示入度或出度的值，纵坐标表示有对应入度或出度值的节点个数。从图 8 可以看出，网络的入度分布出现了严重的“重尾”现象，说明存在部分节点其入度远远大于其它节点的入度。相比之下，出度分布的“重尾”现象就要缓和了许多。另外，无论是入度分布还是出度分布，在双对数坐标系下大体上都呈现出一条直线，说明它们满足“无标度”分布。由此，分别对入度和出度的累积概率分布进行曲线拟合，给出幂律值及拟合标准差。执行网络累积入度和累积出度分布如图 9 所示。

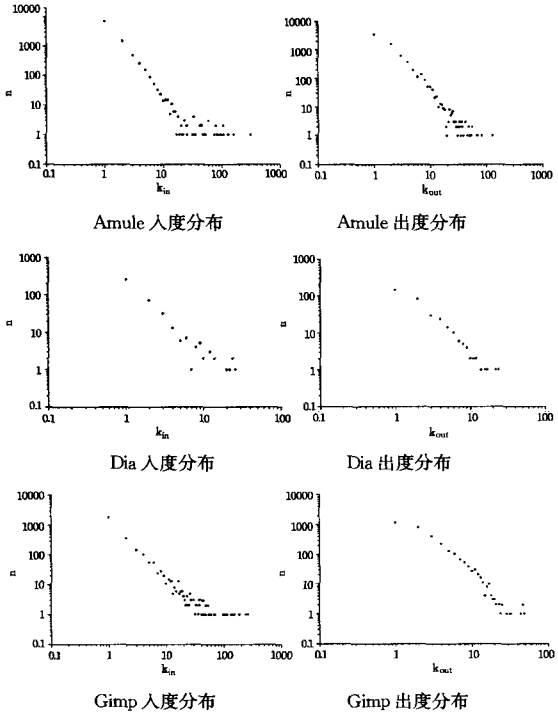


图 8 执行网络入度和出度分布

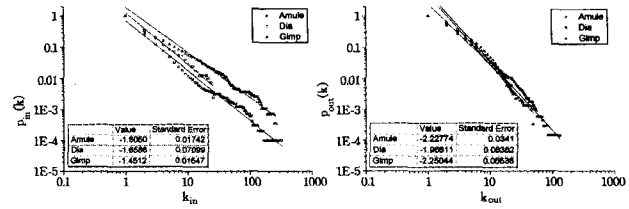


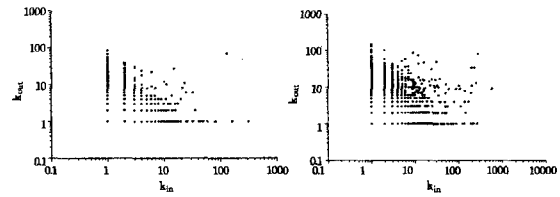
图 9 执行网络累积入度和累积出度分布

从图 9 可以看出，累积入度幂律值要小于累积出度幂律值，换算成标准幂律值后都介于 2～4 的适当幂指数区间(在针对已有的许多无标度分布复杂网络的测量都发现幂指数落在该范围之内，如 Internet 网的幂指数为 2.5、蛋白质网的幂指数为 2.4 等)。

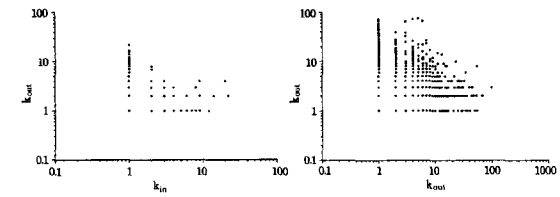
4.6 度相关性

Amule 在平均情况下执行网络的度相关系数为 0.016，静态网中为 0.028，表明入度和出度具有弱的正相关性，即入度大的节点可能出度也大。Dia 的系数值在动态下和静态下分别为－0.128 和－0.077，Gimp 的系数值分别是－0.034 和－0.030，它们的入度和出度表现出弱的负相关性，那些入度大的节点更可能拥有小的出度，入度小的节点出度可能大。另外，相比静态网络，Dia 和 Gimp 执行网络的度相关系数绝对值变大了而 Amule 变小了，但它们都意味着函数网络的入度和出度在执行过程中具有向负相关性变化的动态行为特征。

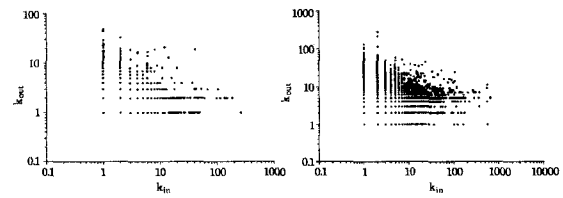
在软件设计中，通常的指导意见是将简单并频繁使用的功能封装到底层函数，然后供其它函数调用来实现相对复杂的功能，这会使得底层函数的入度大但出度小，进而导致网络的入度和出度呈现负相关性。如 Gimp 中的底层函数“gimp_object_get_type”实现了很多模块组件中需要获取对象类型的功能，其入度达到了 261，而出度却仅为 1。图 10 是 3 个软件单个用例的执行网和静态网中节点的入度和出度分布情况。



(a) Amule 执行网络 Amule 静态网络



(b) Dia 执行网络 Dia 静态网络



(c) Gimp 执行网络 Gimp 静态网络

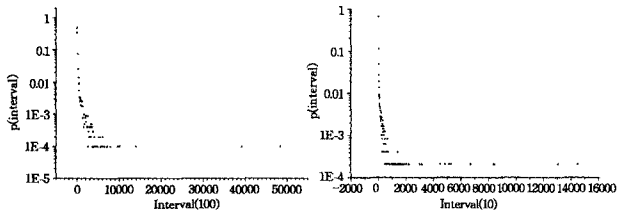
图 10 执行网络和静态网络的节点入度与出度分布

从上图可以看出，在入度和出度双对数坐标系下，Dia 和

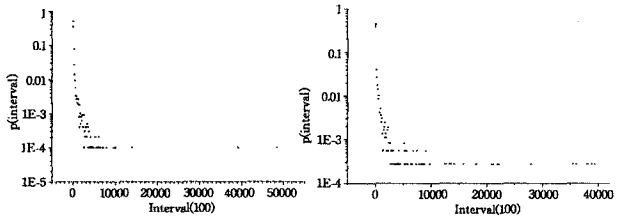
Gimp 在执行网络和静态网络中的节点绝大部分聚集在左下角区域并呈“下三角型”型分布，这是入度和出度具有负相关性的标志。对于 Amule，从执行网络和静态网络都可以发现，确实有少部分节点同时拥有较高的入度和出度，它们处于左“下三角型”的对角区域，使得整体网络的入度和出度具有很弱的正相关性。

4.7 介数

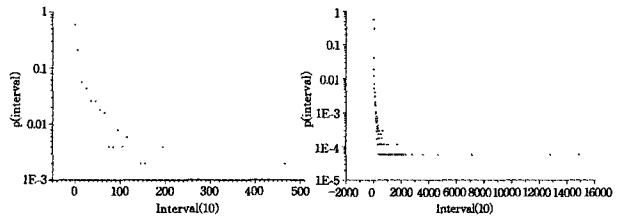
节点的介数反映了对网络中其它节点的交往控制能力。图 11 是 3 个软件单个用例执行网络和静态网络的节点介数值分布情况，其中横坐标表示介数值的区间（分为$(0,m)$，$[m,2m)$，…，以及一个特殊区间$[0,0]$，m 是区间的大小），纵坐标表示节点的介数落在对应区间的概率。



(a) Amule 执行网络　(b) Dia 执行网络



(c) Amule 静态网络　(d) Gimp 执行网络



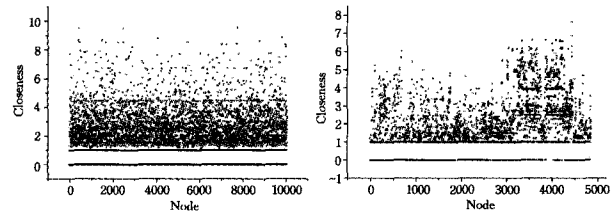
(e) Dia 静态网络　(f) Gimp 静态网络

图 11　单个执行网络和静态网络中的介数分布

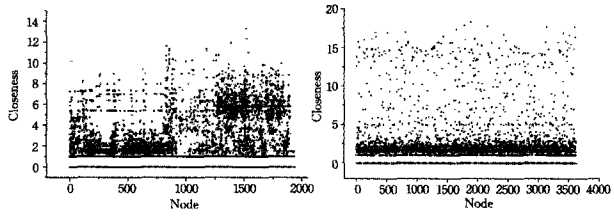
如图 11 所示，3 个软件执行网络中个别节点的介数很大，它们控制着其它节点对的交互，在网络中起着“枢纽”的作用；而其它绝大多数节点相比之下介数都很低，其中介数值为 0 的节点占了相当大的比例，Amule，Dia，Gimp 执行网络中分别为 34.43%，57.76%，39.12%。类似的情况也出现在各软件的静态网络中，其中介数为 0 的节点比例分别是 79.94%，65.84%，57.08%，均高于对应的执行网络，表明实际发生的函数调用网络存在更多拥有控制能力的节点，它们支撑着网络的动态行为，是软件运行过程中系统稳定性的重要保障。

4.8 接近度

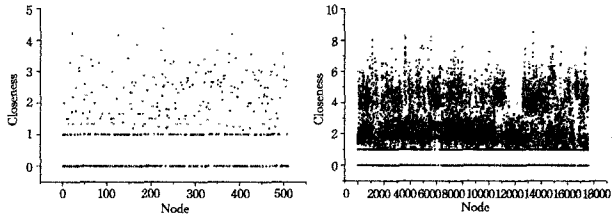
接近度的值越小表明节点在所处的连通组件中离其它节点的距离之和相对越近，越有可能是该连通组件的中心；反之，值越大则越可能是边缘节点。这里所指的“相对”是针对节点出度而言的，若节点的出度为 0，则不存在节点的可达集，在这种情况下其介数也为 0，但这并不表示该节点是连通组件的中心。图 12 是 3 个软件单个用例的执行网络和静态网络的节点接近度分布情况。



Amule 执行网络　Dia 静态网络



Amule 静态网络　Gimp 执行网络



Dia 执行网络　Gimp 静态网络

图 12　执行网络和静态网络中的接近度分布

如图 12 所示，无论是执行网还是静态网都含有相当多接近度为 0 和 1 的节点，它们在图中形成了一条明显的直线。接近度为 0 的节点表明其出度也为 0，不存在可达集。通过查看接近度为 1 的节点，发现绝大部分的可达集中仅存在唯一的元素。在 Amule 的执行网络中，接近度约为 3 的节点也占有一定量的比例。另外一个有趣现象是 Amule 和 Dia 的执行网络中拥有的节点最大接近度比在静态网络中的要小，而比 Gimp 却要大。软件的这种不一致在已经测量的其它结构方面（如网络直径、平均路径长度）也有所表现。

结束语　经过以上的分析可以看出，对于函数网络，无论是静态下的关联网络还是动态下的执行网络，都是一类具有“小世界”和“无标度”特性的异构网络。网络的平均路径长度相对很短，节点连接具有很强的方向单一性，入度和出度服从幂律分布，网络稀疏但却呈现高聚类特点，这些都体现了软件工程中所提倡的“高类聚、低耦合”等重用原则。

但相比于静态网络，以用例驱动的执行网络按需将功能关联的节点动态地绑在了一起，使得网络的密度和聚类系数有所变大，但执行网络并不是简单地选择覆盖静态关联网络，而是还具有诸如节点和边的增加，调用路径重组等行为特性。另外，在执行网络中，节点度的负相关性得以加强，含有入度和出度的节点比例趋于平衡，以及拥有更多控制牵引能力的节点，可能对其在动态运行环境下的网络稳定性起到了一定的作用。

对软件系统在动态执行环境下的结构特征的研究能更加本质、深入地认识这类人造的复杂系统，在度量软件、改善体系结构和保障质量等方面具有十分重要的现实意义。

本文以开源软件为基础，利用复杂网络理论对软件执行期间的函数调用关系进行网络建模和结构特征定义，通过模拟软件在真实环境中的使用情况对该类网络的结构特征进行

了研究。相关结论揭示了软件系统的动态执行特性，对于软件运行维护及质量保障具有重要的指导意义。

参 考 文 献

[1] 岳昆，王晓玲，周傲英. Web 服务核心支撑技术：研究综述[J]. 软件学报，2004，15(3)：428-442

[2] Barabasi A L, Albert R. Emergence of scaling in random networks. Science，1999，286(5439)：509-512

[3] Newman M E J. The structure and function of complex networks[J]. SIAM Review，2003，45：167-256

[4] Valverde S，Ferrer-Cancho R，Sole R. Hierarchical Small Worlds in Software Architecture，Santa Fe Inst[J]. Working Paper，2003：SFI/03-07-044

[5] 马于涛，何克清，李兵，等. 网络化软件的复杂网络特性实证[J]. 软件学报，2011，22(3)：381-407

[6] Myers C. Software system as complex networks：structure，function，and evolvability of software collaboration graphs[J]. Physical Review E 68(2003) 046116，1-046116. 15

[7] de Moura A，Lai Y-C，Motter A，Signature of small-world and scale-free properties in large computer programs[J]. Physical Review E 68(2)(2003) 017102，1-017102. 4

[8] Concas G，Marchesi M，Pinna S，et al. Power-laws in a large object-oriented software system[J]. IEEE Transactions on Software Engineering，2007，33(10)：687-707

[9] 闫栋，邓国宁. 大规模软件系统的无标度特性与演化模型[J]. 物理学报，2006，55：3799-3804

[10] 韩言妮，李德毅，陈桂生. 软件网络的多粒度拓扑特性分析及其应用[J]. 计算机学报，2009，32(9)：1711-1721

[11] Cai K Y，Yin B B. Software Execution Processes as an Evolving Complex Network[J]. Information Sciences，2009，179(12)：1903-1928

[12] Potanin A，Nobble J，Frean M，et al. Scale-free geometry in object-oriented programs[J]. Communication of the ACM，2005 (48)：99-103

(上接第 205 页)

$$\bar{n}=\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}f_{mn}\frac{n}{\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}f_{mn}} \tag{6}$$

式中，$m=1,2,\cdots,M-1$，$n=1,2,\cdots,N-1$；$(\bar{m},\bar{n})$就定义为整个字符的中心位置。定义中心矩如下：

$$L_{i,j}=\sum_{n=(j-1)\frac{N}{2}}^{j\frac{N}{2}}\sum_{m=(i-1)\frac{M}{2}}^{j\frac{M}{2}}(m-\bar{m})(n-\bar{n}) \tag{7}$$

式中，$i=1,2$；$j=1,2$，这样可以得到 4 个重心矩特征：L_{ij} $(i=1,2;j=1,2)$。它们分别反映 4 个象限中像素偏离重心位置的整体特征。本文中选用重心及重心矩特征计算后点阵的重心位置，以及 4 个象限的重心矩，共 5 个特征，包括一个二维特征，4 个一维特征。

2.3 BP 神经网络的学习和训练

从特征的选取过程可以发现，被选取的特征因素很可能存在相互交叉现象。由于多层 BP 网具有自学习、容错性、分类能力强和并行处理等特点，因此用它来训练与识别手写体数字比较合适。文中使用的 BP 网络分为 3 层，其输入层有 6 个点，分别对应特征矢量的 6 个分量，其输出层有 10 个输出点。通过对数字样本进行训练，将学习的结果纳入到聚类源。

3 结果分析

表 1 实验结果

样本输入	正确率	误识率	拒识率
测试样本 1(印刷体)	95.50%	2.50%	2.0%
测试样本 2(手写体)	90.10%	7.20%	2.70%
测试样本 3(手写体)	93.10%	5.30%	1.60%
测试样本 4(手写体)	95.20%	4.10%	0.70%
测试样本 5(手写体)	92.60%	2.50%	4.90%

提取测试片断，按上述方法对每个手写体数字提取特征矢量，得到 5 个测试样本，每个样本有 115 个待识别数字，每个数字的特征矢量为 6 位。对于这个输入样本，经过上述 BP 网络学习以后，将得到一个十维的输出向量。在该 BP 网络中，选择 η 的初始值为 0. 15，然后用时间退火函数让 η 值慢慢减小。最后，选择定势态因子 α 为 0. 175，学习结束的条件为网络的均方根 $\Delta E=0.001$。实验结果见表 1 及图 3 所示。

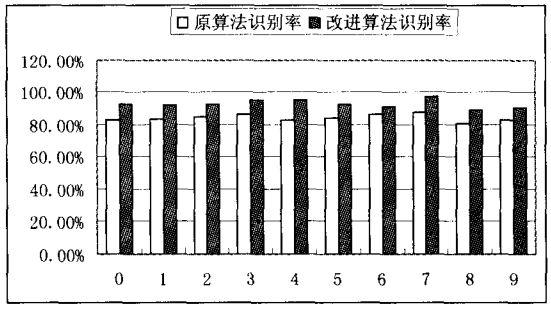


图 3 两种 BP 算法分别对样本中 0～9 的识别率

参 考 文 献

[1] 刘勇，赵斌，夏绍玮. 模糊超椭球分类算法及其在无约束手写体数字识别中的应用[J]. 清华大学学报：自然科学版，2000，40(9)：120，124

[2] Demirekler M，Altincy H. Plurality Voting-based Multiple Classifier Systems：Statistically Independent with Respect to Depend Classifier Sets[J]. Pattern Recognition，2002，35：2365-2379

[3] Altineay H，Demirelder M. Undesirable Effects of Out-put Normalization in Multiple Classifier Systems[J]. Pattern Recognition Letters，2003，24：1163. 1170

[4] Bezdek J. Pattern Recognition with Fuzzy Objective Function Algorithms[M]. NewYork：Plenum Press，1981

[5] 胡瑞敏. 广义神经网络理论和信号识别系统的研究[D]. 武汉：华中理工大学，1994，9

[6] 王爱国，肖定中. 一种改进的 BP 神经网络模型及其在语音识别中的应用[J]. 光通信研究，1996

[7] 陈敏，刘君. BP 网络的改进及其应用[J]. 湖南文理学院学报，2005

[8] 戚建宇，何松. 改进 BP 神经网络的普通话单字发音标准度研究[J]. 福建电脑，2009

[9] 董慧. 手写体数字识别中的特征提取和特征选择研究[D]. 北京：北京邮电大学，2007