# 基于有向带权图迭代的面向对象系统分解方法*

罗 景+, 赵 伟, 秦 涛, 姜人宽, 张 路, 孙家骕

(北京大学 信息科学技术学院 软件研究所,北京 100871)

# A Decomposition Method for Object-Oriented Systems Based on Iterative Analysis of the Directed Weighted Graph

LUO Jing+, ZHAO Wei, QIN Tao, JIANG Ren-Kuan, ZHANG Lu, SUN Jia-Su

(Institute of Software, School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China)

+ Corresponding author: Phn: +86-10-51605389, E-mail: luoj@sei.pku.edu.cn, http://eecs.pku.edu.cn/

Received 2003-10-24; Accepted 2004-02-05

**Luo J, Zhao W, Qin T, Jiang RK, Zhang L, Sun JS. A decomposition method for object-oriented systems based on iterative analysis of the directed weighted graph. *Journal of Software*, 2004,15(9):1292~1300.**
http://www.jos.org.cn/1000-9825/15/1292.htm

**Abstract**: Aiming at the problem of how to acquire components from existing systems, this paper proposes a decomposition method for object-oriented systems based on iterative analysis of the directed weighted graph. This method uses the directed weighted graph as the representation of object-oriented systems, and an iterative algorithm for analyzing the independence of sub-graphs at different granularity levels. Those highly independent ones are chosen as candidate components. Experimental results show that this method is effective and can improve the existing decomposition methods in terms of accuracy.

**Key words**: software component; component acquirement; directed graph analysis; cohesion and coupling; independence measurement

**摘 要**: 针对如何从现存的系统中提取构件的问题,提出了一种基于有向带权图迭代分析的面向对象系统分解方法.它将面向对象系统抽象为一个有向带权图,使用迭代算法考察不同粒度的子图的独立性,并选择独立性高的作为候选构件.实验结果表明,该方法是一种有效的系统分解方法,在准确性上比现有系统分解方法有所提高.

**关键词**: 软件构件;构件提取;有向图分析;内聚耦合;独立性度量

**中图法分类号**: TP311 **文献标识码**: A

* Supported by the National High-Tech Research and Development Plan of China under Grant No.2001AA113070 (国家高技术研究发展计划(863)); the National Grand Fundamental Research 973 Program of China under Grant No.2002CB31200003 (国家重点基础研究发展规划(973))

**作者简介**: **罗景**(1980－),男,湖南隆回人,硕士生,主要研究领域为软件工程,程序理解;**赵伟**(1977－),女,博士生,主要研究领域为软件维护,程序理解,逆向工程;**秦涛**(1978－),男,硕士,主要研究领域为程序理解,软件工程;**姜人宽**(1980－),男,硕士生,主要研究领域为程序理解;**张路**(1973－),男,博士,副教授,主要研究领域为软件工程,程序理解,配置管理,人工智能;**孙家骕**(1946－),男,教授,博士生导师,主要研究领域为计算机语言及编译技术,软件工程,软件逆向工程.

基于构件的软件开发是解决软件危机的一种现实而有效的途径[1],它利用构件的可复用特性减少了软件开发中的重复劳动.对于面向对象系统而言,构件可以是类、类树、类簇,甚至是一个由众多类组成的框架[2].一般来说,构件有两种获取途径:有目的的构件生产以及从现存的系统中提取构件[1].由于目前存在大量正在使用的面向对象系统,从它们中提取构件已成为学术界和产业界关注的焦点.这种构件提取主要包括以下一些活动[3]:首先,需要对整个系统进行分解,得到候选构件;其次,需要对候选构件进行可复用性度量,筛选出可复用性高的候选构件;最后,可能需要对这些构件进行适当修改,进一步提高其可复用性.本文主要探讨如何对系统进行分解,得到候选构件.

从现有文献来看,面向对象系统分解方法大致可分为两类:知识匹配和结构分析.知识匹配方法以系统中目标构件的相关知识为驱动,将这些知识与系统中的成分进行匹配,相匹配的成分就成为候选构件.这类方法的代表是文献[4,5].它们的一个主要缺点是:匹配中所需知识的获取和表示在实践中通常比较困难.结构分析方法将软件系统抽象为可以数学化表示的结构,比如树[6]和图[3,7],然后对抽象出来的结构进行分析,将整个结构分解为多个子结构,每个子结构对应一个候选构件.

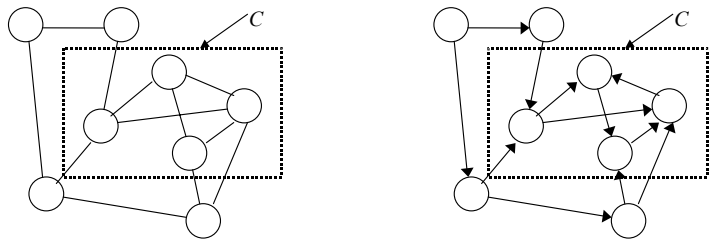
本文针对面向对象的软件系统,在总结现有图分析方法存在的问题的基础上,提出了一种基于有向带权图迭代分析的系统分解方法.我们的方法将面向对象软件系统抽象为一个有向带权图,以迭代的方式逐步考察图中不同粒度的子图,并通过独立性度量选取独立性高的子图作为构件候选者.在迭代过程中,每次迭代得到的候选构件将被抽象为下一次迭代的图中的一个顶点.从我们的实验结果看,我们的方法在准确性上比现有的图分析方法有所改进.

## 1 现有的图分析方法及存在的问题

由于图比树更能充分表达面向对象系统的结构信息,因而图分析方法比树分析方法更得到学术界的重视.文献[3,7]是图分析方法的主要代表.这两种方法都是将面向对象系统抽象为无向图,其中类抽象为图中的顶点,而图中的边表示类间的关系;然后通过特定的算法计算出类间的关系的度量值,并删除那些度量值小于某个阈值的关系;剩下的每个连通子图都作为一个候选构件.这两种方法的主要区别是采用不同的度量值计算方案以及不同的阈值.综合考察这两种方法,我们发现存在以下一些问题:

- 关系的方向性

在面向对象软件系统中,各个类之间的关系都存在方向性[8].继承关系表示子类继承了父类的方法与属性,子类依赖于父类而存在,而父类并不一定依赖于子类,因此继承关系是由子类指向父类具有方向性的一种关系;整体部分关系又称为聚合关系,它把一组具有整体部分关系的类组织在一起,整体类依赖于部分类,而部分类不一定依赖于整体类;实例连接关系描述了对象间的静态联系,一般而言,类 $A$ 的对象引用了类 $B$ 的对象表明类 $A$ 可能依赖于类 $B$,但类 $B$ 不依赖于类 $A$;消息连接关系描述对象之间的动态联系,由这种关系带来的类间的依赖性也是有方向性的,即消息发送对象的类可能依赖于消息接收对象的类.在进行系统分解时,如果将系统抽象为无向图,就难以有效地识别出那些被其他部分依赖,但自身独立性良好的构件.如图 1 所示.




(a) Undirected graph based analysis (b) Directed graph based analysis

(a) 无向图分析 (b) 有向图分析

Fig.1 Comparison of analysis based on undirected and directed graphs

图 1 无向图与有向图分析的对比

如果把系统抽象为无向图(如图 1(a)所示),则 $C$ 可能因为与系统的其他部分关联过于紧密而不能作为候选对象;如果把系统抽象为有向图(如图 1(b)所示),则可能发现实际上 $C$ 与系统其他部分的关联都是其他部分依赖于 $C$,而 $C$ 本身是独立的,因而还是可以作为候选构件的.

- 关系的类型

如前所述,面向对象系统中类之间存在着不同种类的关系,而这些关系在整个结构上所起的作用也是不一样的.例如,类 $B$ 从类 $A$ 继承,$B$ 的很多方法和属性可能都是从 $A$ 继承而来,因而 $B$ 很难独立于 $A$ 而存在,在系统分解时,如果 $A$ 和 $B$ 分在两个候选构件里,$B$ 所在的候选构件就可能会由于紧密依赖于 $A$ 所在的候选构件,而自身的可复用性较低.如果 $A$ 和 $B$ 之间不具有继承关系,而是具有消息连接关系,把 $A$ 和 $B$ 分在不同的候选构件里并不一定会导致这两个候选构件的耦合过于紧密,因而不会对可复用性有太大的影响.如果在进行系统分解时能够充分考虑到不同的关系所起的作用也不同,就有可能进一步提高分解的准确性.在现有的文献中,文献[3]已对关系的类型进行了区分,而文献[7]并没有考虑关系类型的区别.

- 系统的构造性

对于一个良构的软件来说,整个系统应该是具备构造性的.也就是说,整个系统可以划分为若干子系统,而子系统又可以划分为更小的子系统,这样一直划分下去,直至每个最小的子系统是一个类.在进行系统分解时,我们完全可以假定系统是良构的,因为我们的目标是提取构件,而从一个非良构的系统中提取构件的意义不大.对于一个具备构造性的系统来说,我们就可以采取自顶向下或自底向上的方式,通过多次迭代来完成系统的划分.如果采用自顶向下的方式,可以首先分解出一些大粒度的候选构件,然后再逐步分解出更小粒度的候选构件;如果采用自底向上的方式,可以首先分解出一些小粒度的候选构件,然后以它们为基础进一步得到更大粒度的候选构件.现有的图分析方法没有充分考虑到系统的构造性,它们需要对整个系统结构进行分析,直接分解出所有候选构件,因而往往会导致分解的准确性有所欠缺.

## 2 有向带权图迭代分析方法

基于以上分析,我们提出了一种基于有向带权图的迭代分析方法.该方法将面向对象系统抽象为有向带权图,顶点代表类,边代表类间的关系,边的方向体现关系的方向,针对关系的不同类型,每条边赋予不同的权.这种类图结构的提取主要采用文献[9]中的工具来完成,并在此基础上将实现中的类间关系映射到本文使用的类间关系:继承关系(inheritance)保持不变,嵌套对象映射为整体部分(whole-part)关系,嵌套指针映射为实例连接(instance-link),依赖关系映射为消息连接(message-link).根据得到的类图表示,在系统分解时,采取自底向上迭代分析的方案.

### 2.1 面向对象系统的有向带权图表示

一个面向对象系统可以抽象为一个有向图 $G=(V,E)$,其中 $V$ 是图中顶点的集合,每个顶点代表一个类;$E=\{\langle v_1,v_2,t\rangle|v_1,v_2\in V,t\in T\}$ 表示图中边的集合,其中 $T=\{\text{inheritance,whole-part,instance-link,message-link}\}$ 是类间关系的集合.对于 $E$ 中的任意一条边 $e=\langle v_1,v_2,t\rangle$,我们定义权值函数 $W(e)=F(t)$,其中函数 $F$ 的定义如下:

$$F(t)=\begin{cases}W_{ih}\ (t=\text{inheritance})\\ W_{wp}\ (t=\text{whole-part})\\ W_{il}\ (t=\text{instance-link})\\ W_{ml}\ (t=\text{message-link})\end{cases}\tag{1}$$

从式(1)可以看出,每条边的权值只与边的类型有关,继承关系的权为 $W_{ih}$,整体部分关系的权为 $W_{wp}$,实例连接关系的权为 $W_{il}$,消息连接关系的权为 $W_{ml}$.

### 2.2 系统分解算法

针对上述面向对象系统的有向带权图表示,我们提出如下系统分解算法:首先根据图的大小确定每次迭代考察的子图的最大规模 $k$,然后对图进行迭代分析,第 1 次迭代的输入是表示整个系统的图.每次迭代考察输入的图的所有规模小于 $k$ 的连通子图,采用独立性度量函数(定义见第 2.3 节)计算每个子图的独立性,选取其中独

立性高于阈值 $d$ 的子图作为候选子图(对应于候选构件),然后将候选子图抽象为一个顶点,并对输入的图进行重构,形成一个规模较小的新图,以新图作为下一次迭代的输入,重复进行迭代直到重构得到的新图的规模小于 $s$ 为止.算法的伪码表示如下:

**算法**.

输入:面向对象系统的有向带权图表示 $G$,最大子图规模 $k$,独立性阈值 $d$ 以及输入图最小规模 $s$.

输出:候选子图集合 Candidates.

```
Candidates=∅
While (sizeOf(G)≥s)
 Begin
      subGraphList=getConnectedSubGraphs(G,k)
      newCandidates=getIndependentSubGraphs(subGraphList,d)
      Candidates=Candidates∪newCandidates
      G=constructNewGraph(G,newCandidates)
 End
```

在上述算法中,使用的函数定义如下:

• sizeOf($G$)的返回值就是 $G$ 中顶点的个数$|V|$.

• getConnectedSubGraphs($G$,$k$)返回 $G$ 中所有规模小于 $k$ 的连通子图.

• getIndependentSubGraphs(subGraphList,$d$)针对 subGraphList 中每个子图 $SG$,计算 $SG$ 的独立性,最后返回所有独立性高于 $d$ 的子图.子图独立性的计算见第 2.3 节.

• constructNewGraph($G$,newCandidates)针对 newCandidates 中的每个候选子图,将其在 $G$ 中收缩为一个顶点,当 newCandidates 中所有子图都已收缩为一个顶点后,得到的新图即为返回值,在新图重构的过程中,如果有两个候选子图存在交集,则选取度量值更高的子图,这对于构件提取是不会构成影响的,因为每一次分析都输出高于阈值的连通子图集合,而且这种方法使得在后续的迭代过程中会将度量值较小的子图中的节点陆续地加入到度量值较高的子图中,作为新的候选子图进行考虑.

设 $G=(V,E)$为一有向带权图,$G_1=(V_1,E_1)$是 $G$ 的一个子图($V_1\in V$ 且 $E_1\in E$),则在 $G$ 中将 $G_1$ 收缩为一个顶点后得到的新图 $G'=(V',E')$的定义如下:

(1) $V'=(V-V_1)\cup\{v\}$,其中 $v\notin V$ 且 $v\notin V_1$,表示收缩后的顶点.

(2) $E'=\{\langle v_1,v_2,t\rangle|\langle v_1,v_2,t\rangle\in E$ 且 $v_1,v_2\in V-V_1\}\cup\{\langle u,v,t\rangle|v$ 为收缩后的顶点,$u\in V-V_1$ 且$\exists v_1\in V_1,\langle u,v_1,t\rangle\in E\}\cup\{\langle v,u,t\rangle|v$ 为收缩后的顶点,$u\in V-V_1$ 且$\exists v_1\in V_1,\langle v_1,u,t\rangle\in E\}$.

(3) 在得到的新图中,如果有 $n$ 条边的顶点与方向相同,则将其抽象为一条权值为 $n$ 边权值之和且方向不变的边.

图 2 是使用该算法对输入进行一次迭代的示例.

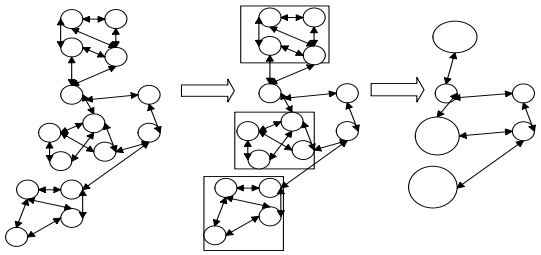


Fig.2 Example of one iteration

图 2 一次迭代的示例

### 2.3 独立性度量

为了在迭代分析中得到独立性比较高的子图,我们设计了一种子图独立性的度量方案,它借鉴了 QMOOD

度量模型中的系统划分质量度量[10].对于图 $G=(V,E)$的一个划分 $C=(G_1,G_2,\ldots,G_n)$,其中 $G_i=(V_i,E_i)(1\le i\le n)$是 $G$ 的一个子图,其质量由下面的公式定义:

$$MQ(C,G)=\frac{\sum_{i=1}^{n}s(G_i,G_i)}{n}-\frac{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}s(G_i,G_j)}{n(n-1)/2} \tag{2}$$

在式(2)中,$s(G_i,G_i)$表示子图 $G_i$的内聚性,$s(G_i,G_j)$表示子图 $G_i$和子图 $G_j$间的耦合性.由此可以看出,划分中每个子图的内聚性越高则划分的质量越高,而子图间的耦合性越高则划分的质量越低.

在此基础上,我们在定义子图的独立性时主要考虑以下一些因素:① 子图本身的内聚性;② 子图与其补图的耦合性;③ 子图的规模.这样,如图 3 所示,我们得到了子图独立性的度量树.

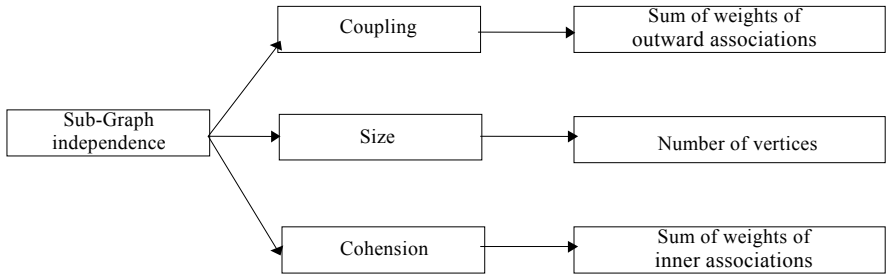



Fig.3 Metric tree for independence

图 3 独立性度量树

从这个度量树可以看出,子图本身的内聚性通过计算子图内部所有关联的权值总和得到;子图与其补图的耦合性通过计算从子图到其补图的所有关联的权值总和得到;子图的规模由子图中顶点的数目确定.其中,关系的方向性在计算耦合性时起着至关重要的作用.将这 3 方面的因素结合起来,我们得到了如下独立性度量公式.对于有向带权图 $G=(V,E)$及其子图 $G'=(V',E')$,从 $G'$到其补图的关系集合为 $O=\{\langle u,v,t\rangle|\langle u,v,t\rangle\in E$ 且 $u\in G'$, $v\in V-V'\}$,子图 $G'$的独立性 IM 定义为

$$IM(G',G)=\frac{\sum_{e\in E'}W(e)}{|V'|\sum_{e\in O}W(e)} \tag{3}$$

从式(3)可以看出,一个子图的内聚性越好,独立性越强,内聚性与独立性成正比;耦合性越强,则独立性越差,耦合性与独立性成反比;子图的规模与独立性成反比.如果子图的耦合性为 0,则给定一个大的度量性度量值,说明这是一个独立性极强的模块.

### 2.4 算法的复杂度分析

对于一个有向带权图 $G=(V,E)(|V|=n,|E|=m)$,下面分析一下用上述算法进行处理的时间复杂性.由于每次迭代会将规模小于 $k$ 的候选构件收缩为一个顶点,所以迭代的次数应为 $O(\log_k n)$.在一次迭代中,规模小于 $k$ 的连通子图的数目在最坏情况下是 $O(n^k)$,因而生成这些子图的复杂性也是 $O(n^k)$;在计算一个子图的独立性时,由于可能要遍历所有的边,因而复杂性是 $O(m)$,即计算所有子图的独立性的复杂度为 $O(mn^k)$.类似地,图的重构的复杂性也是 $O(mn^k)$.这样,整个算法的时间复杂性是 $O(mn^k\log n)$.实际上,由于迭代过程中图的规模会迅速减小,一次迭代的复杂性也会迅速降低,$O(mn^k\log n)$只是时间复杂性的一个上界,而不是准确的复杂性.

从上面的复杂性分析可以看出,整个算法的复杂性与 $k$ 是指数相关的,因而在实际应用时 $k$ 只能取一些相对较小的值.事实上,如果假设待分解的系统是良构的,$k$ 也不需要取很大的值,因为大的构件也是由小的构件构成的,每次迭代时不需要考虑太大的构件.另外,由于 $G$ 中每个顶点代表一个类,顶点的个数也会是比较有限的.一般地,一个比较大的系统也只有几百个或最多上千个类.事实上,如果我们把 $k$ 的值设为 4~5,用目前的微机就完全可以处理通常遇到的各种面向对象系统问题.

## 3 系统原型的实现

基于上述方法,我们实现了一个原型系统.如图 4 所示,该系统利用青鸟程序理解工具[9]作为软件系统信息获取器,获得软件系统的类图结构;针对类图结构,我们设计了一个有向带权图的转换器,将类图结构转换为有向带权图;在有向带权图的基础上,用一个有向带权图的分析器实现上述分解算法,从有向带权图中分解出各个层次具有良好独立性的子图;最后利用子图与代码的对应关系通过代码抽取器将候选子图对应的候选构件抽取出来.

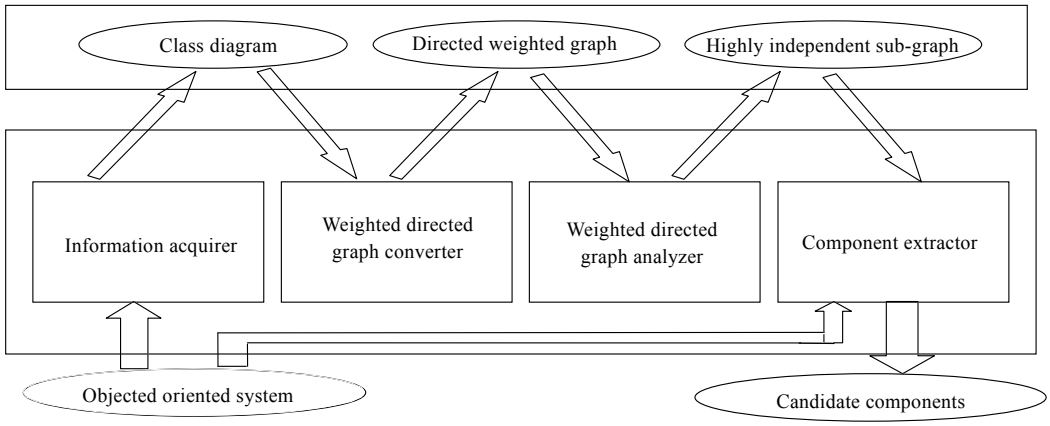



Fig.4 A candidate component extraction system based on analysis of the directed weighted graph

图 4 一个基于有向带权图分析的候选构件抽取系统

## 4 实例研究

我们在系统原型的基础上,针对一个加密解密工具[11]进行了实例研究.该工具有 17 个类,大约 12 000 行代码,采用混合的加密解密方法,并提供了一个良好的界面.为了验证该方法的有效性,针对实例,我们同时采用有向迭代方法、文献[7]中的方法、无向迭代方法以及人工方法进行了实验分析,并对结果进行了比较研究.在有向迭代方法中,确定继承关系的权 $W_{ih}$ 为 0.9,整体部分关系的权 $W_{wp}$ 为 0.8,实例连接的权 $W_{il}$ 和由消息连接的权 $W_{ml}$ 均为 0.6.该工具的有向图表示如图 5 所示(为了清晰起见省略了权).

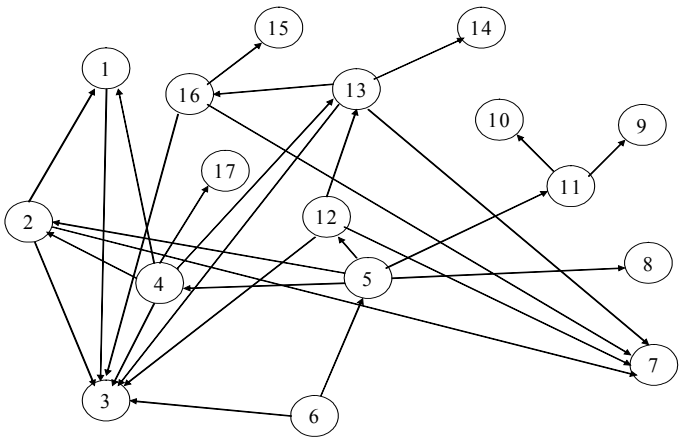



Fig.5 The directed graph for the studied tool

图 5 目标工具的有向图表示

### 4.1 有向带权迭代分析与文献[7]中方法的比较

表 1 是通过人工分析得到的结果.

**Table 1** Results of manual analysis

**表 1** 人工分析结果

| Candidate component number | Class number | Corresponding function |
|---|---|---|
| 1 | 1,3 | File input and output |
| 2 | 3,7,16 | Big prime number generation |
| 3 | 9,10,11 | About dialogue |
| 4 | 1,2,3,7 | DES encrypt and decrypt algorithm |
| 5 | 3,7,13,14,16 | RSA encrypt and decrypt algorithm |

采用我们的方法得到的独立性最高的 10 个候选构件如下:{**1**,**3**},{**9**,**10**,**11**},{**3**,**7**,**16**},{**1**,**2**,**3**,**7**},{1,3,7,16},{1,3,9,10,11},{1,2,3,7,16},{**3**,**7**,**13**,**14**,**16**},{1,3,7,13,14,16}和{3,7,12,13,14,16}.由此可以看出,人工分析得到的候选构件用有向带权迭代方法都可以得到,同时,它还得到了在结构上独立而在语义上不独立的候选构件,可以通过语义分析排除.而采用文献[7]中的方法得到的候选构件如下:{6},{5,8},{**9**,**10**,**11**}和{1,2,3,4,7,12,13,14,15,16,17}.其中只有一个候选构件与人工分析的结果相符合.

从以上结果可以看出,文献[7]中的方法由于不能处理构件嵌套以及构件共享某些类的情况,因而只能准确分解出少量构件.同时,由于这种方法没有考虑关系的方向性,造成了一些错误的分解.我们的方法能够有效地解决以上两种情况,因而能够得到较好的结果.

### 4.2 有向带权迭代分析与无向带权迭代分析比较

图 6 显示了用有向带权迭代和无向带权迭代分别分析实例系统得到的独立性最好的 100 个子图的独立性比较.从图中可以看出,有向带权迭代分析方法得到的子图独立性普遍地高于无向带权迭代分析结果,这是由于在有向带权迭代分析中考虑了边的有向性的结果.

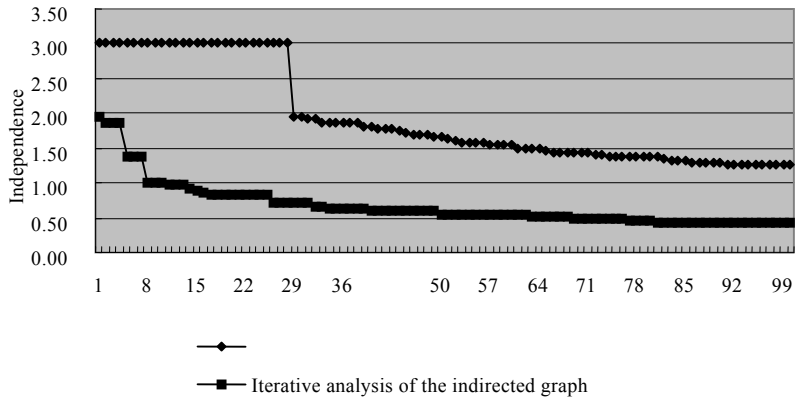



Fig.6 Comparison of independence of candidate sub-graphs

图 6 候选子图独立性的比较

图 7 显示了这两种方法得到的独立性最好的 100 个候选子图在不同子图规模上的分布.在有向带权迭代方法得到的子图中,大部分子图的规模集中在 5~9 这个相对于 17 适中的规模之间.在无向带权迭代分析得到的这些子图中,大部分子图的规模集中在 11~15,对于一个 17 类的系统,这些子图规模偏大,不易于作为构件候选.产生这种情况的原因在于:在无向迭代分析中,一个子图与其补图的耦合性主要取决于补图的规模,因为它与补图之间的任何关系都会增大耦合性,只有减小补图规模,耦合性才会减小.

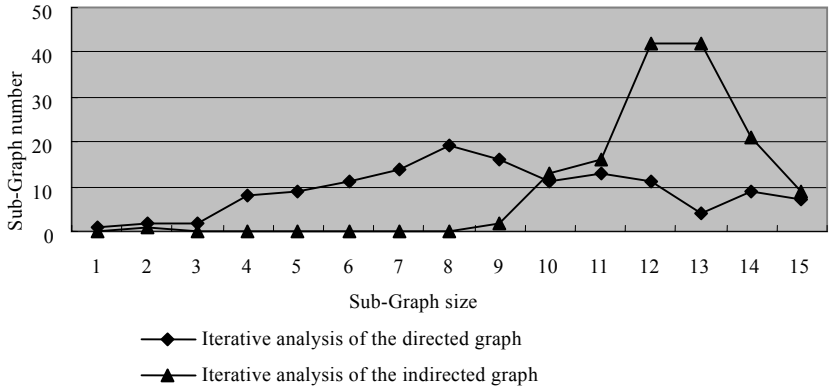



Fig.7 Comparison of distribution of candidate sub-graphs

图 7 候选子图分布的比较

## 5 结束语

从现存面向对象系统中提取构件是支持基于构件的软件开发的重要手段,而有效的系统分解是构件提取的关键技术之一.本文在分析现有面向对象系统分解方法的基础上,提出了一种基于有向带权图迭代分析的面向对象系统分解方法.这种方法考虑了类之间关系的类型和方向性,并充分利用了系统的构造性,因而能有效地对面向对象系统进行分解,为构件提取奠定基础.在我们的实验中,这种方法的准确性比现有方法有所提高.

**References**:

[1] Yang FQ, Mei H, Li KQ. Software reuse and software component technology. Acta Electronica Sinica, 1999,27(2):68~75 (in Chinese with English abstract).

[2] Wu Q. Research on the component composition technology [Ph.D. Thesis]. Beijing: Peking University, 1998 (in Chinese with English abstract).

[3] Zhou X, Chen XK, Sun JS, Yang FQ. Software measurement based reusable component extraction in object-oriented system. Acta Electronica Sinica, 2003,31(5):649~653 (in Chinese with English abstract).

[4] Spinellis D, Raptis K. Component mining: A process and its pattern language. Information and Software Technology, 2000,42(9): 609~617.

[5] Pinzger M, Gall H. Pattern-Supported architecture recovery. In: Proc. of the 10th Int'l Workshop on Program Comprehension. Paris: IEEE Computer Society Press, 2002. 53~61.

[6] Biggerstaff T, Mitbander B, Webster D. The concept assignment problem in program understanding. In: Proc. of the Int'l Conf. on Software Engineering. Baltimore: IEEE Computer Society Press, 1993. 482~498.

[7] Chiricota Y, Jourdan F, Melancon G. Software components capture using graph clustering. In: Proc. of the 11th IEEE Int'l Workshop on Program Comprehension. Portland: IEEE Computer Society Press, 2003. 217~226.

[8] Shao WZ, Yang FQ. Object-Oriented System Analysis. Beijing: Tsinghua University Press, 1998 (in Chinese).

[9] Zhou X, Sun JS, Yang FQ. The Jade Bird C++ program comprehension tool. Computer Engineering, 2000,26(11):80~81 (in Chinese with English abstract).

[10] Mitchell BS, Mancoridis S. Comparing the decompositions produced by software clustering algorithms using similarity measurements. In: Proc. of IEEE Int'l Conf. on Software Maintenance. Florence: IEEE Computer Society Press, 2001. 744~753.

[11] VCBase. A mixed encryption system. 2002. http://www.vckbase.com/code/downcode.asp?id=1781 (in Chinese).

**附中文参考文献**:

[1] 杨芙清,梅宏,李克勤.软件复用与软件构件技术.电子学报,1999,27(2):68~75.

[2] 吴穹.构件组装技术研究[博士学位论文].北京:北京大学,1998.

[3] 周欣,陈向葵,孙家骕,杨芙清.面向对象系统中基于度量的可复用构件获取机制.电子学报,2003,31(5):649~653.

[8] 邵维忠,杨芙清.面向对象的系统分析.北京:清华大学出版社,1998.

[9] 周欣,孙家骕,杨芙清.青鸟 C++程序理解工具.计算机工程,2000,26(11):80~81.

[11] VC 知识库.混合密码系统.2002.http://www.vckbase.com/code/downcode.asp?id=1781

## 敬 告 作 者

《软件学报》创刊以来,蒙国内外学术界厚爱,收到许多高质量的稿件,其中不少在发表后读者反映良好,认为本刊保持了较高的学术水平.但也有一些稿件因不符合本刊的要求而未能通过审稿.为了帮助广大作者尽快地把他们的优秀研究成果发表在我刊上,特此列举一些审稿过程中经常遇到的问题,请作者投稿时尽量予以避免,以利大作的发表.

1. 读书偶有所得,即匆忙成文,未曾注意该领域或该研究课题国内外近年来的发展情况,不引用和不比较最近文献中的同类结果,有的甚至完全不列参考文献.

2. 做了一个软件系统,详尽描述该系统的各个方面,如像工作报告,但采用的基本上是成熟技术,未与国内外同类系统比较,没有指出该系统在技术上哪几点比别人先进,为什么先进.一般来说,技术上没有创新的软件系统是没有发表价值的.

3. 提出一个新的算法,认为该算法优越,但既未从数学上证明比现有的其他算法好(例如降低复杂性),也没有用实验数据来进行对比,难以令人信服.

4. 提出一个大型软件系统的总体设想,但很粗糙,而且还没有(哪怕是部分的)实现,很难证明该设想是现实的、可行的、先进的.

5. 介绍一个现有的软件开发方法,或一个现有软件产品的结构(非作者本人开发,往往是引进的,或公司产品),甚至某一软件的使用方法.本刊不登载高级科普文章,不支持在论文中引进广告色彩.

6. 提出对软件开发或软件产业的某种观点,泛泛而论,技术含量少.本刊目前暂不开办软件论坛,只发表学术文章,但也欢迎材料丰富,反映现代软件理论或技术发展,并含有作者精辟见解的某一领域的综述文章.

7. 介绍作者做的把软件技术应用于某个领域的工作,但其中软件技术含量太少,甚至微不足道,大部分内容是其他专业领域的技术细节,这类文章宜改投其他专业刊物.

8. 其主要内容已经在其他正式学术刊物上或在正式出版物中发表过的文章,一稿多投的文章,经退稿后未作本质修改换名重投的文章.

本刊热情欢迎国内外科技界对《软件学报》踊跃投稿.为了和大家一起办好本刊,特提出以上各点敬告作者.并且欢迎广大作者和读者对本刊的各个方面,尤其是对论文的质量多多提出批评建议.