一种基于聚集系数的局部社团划分算法

李孔文 顾 庆 张 尧 陈道蓄
(南京大学计算机科学与技术系 软件新技术国家重点实验室 南京 210093)

摘 要 社团划分算法是复杂网络研究中的一个热点问题。传统的复杂网络社团划分算法都必须获得全局网络的信息。随着网络规模不断增大,获得全局信息的难度随之增加;而在很多情况下只关心网络中某节点所在的局部社团。为了准确、快速地找到大规模复杂网络中的局部社团,提出了一种基于节点聚集系数性质的局部社团划分算法。该算法根据节点的连接频度,利用节点聚集系数的性质,从网络中某一待求节点开始,通过搜索邻居节点,划分该节点的社团结构。该算法只需要了解与待求节点相关的局部网络信息,在解决局部社团划分问题时其时间复杂度比传统的社团划分算法低。同时,该算法也可以应用于复杂网络全局社团结构的划分。利用该算法分别对 Zachary 空手道俱乐部网络和由 Java 开发工具包构成的软件网络图进行社团划分实验,并且分别对实验结果与对象网络的具体特征进行了对比分析。

关键词 局部社团,聚集系数,社团划分

Local Community Detecting Method Based on the Clustering Coefficient

LI Kong-wen GU Qing ZHANG Yao CHEN Dao-xu
(State Key Laboratory of Novel Software Technology, Department of Computer Science and Technology, Nanjing University, Nanjing 210093, China)

Abstract Community detecting has been a research topic in the complex network area. The global information of the whole network, which is required by the traditional community detecting algorithms, is hard to get when the scale of the network grows. On the other hand, in many cases we only care about the local community of one particular node of the network. To make the local community detecting faster and more accurate, this paper proposed a local community detecting method based on the clustering coefficient of the nodes. The proposed method, which leverages the connectivity density and the characteristics of clustering coefficient, starts from the target node of the network and detects the community it belongs to by searching the neighbor nodes. This method requires only the local network information related to the target node and is faster compared to the traditional community detecting algorithm. It is also applicable for global community structure detecting. The method was applied to the Zachary network and JSCG, and the experiment results were analyzed by comparing with the actual characteristics of the object network.

Keywords Community, Clustering coefficient, Community detecting

1 引言

复杂网络可以用来描述社会关系网络、生物网络、通信网络、网页链接关系等,但目前复杂网络还没有精确严格的定义。它既不是规则网络,也不是随机网络,而是具有与两者皆不相同的统计特征的网络,例如小世界效应[1]和无标度特性[2]。随着对复杂网络性质的物理意义和数学特性的深入研究,研究者们发现很多实际网络中都具有一个共同性质:网络中存在社团结构,即网络由若干个"社团"组成,社团内部的节点之间连接紧密,社团之间的连接稀疏[3]。

如何在复杂网络中进行正确的社团划分成为当前复杂网络研究中的一个热点。为了寻找复杂网络中的社团结构,研究人员提出了很多划分社团的算法,例如谱平分算法、GN 算法[6]、快速 Newman 算法[7]等都是基于网络的全局信息揭示网络的社团结构。然而,当网络规模过于庞大时,获得全局信息非常困难,特别是不断动态变化的网络,如互联网;另外,在很多情况下,研究人员关注的是网络的局部社团结构。例如,在社会网络中搜索时通常只关心某个人所在的社团,而不需要了解整个社会网络的社团结构。或者在图书销售关系网络中,购书者只需了解某书相关主题的书籍,即某书所在的图书社团。在这些情况下,就不需要耗时寻找网络的全局社团结构,而只需搜索网络中某个节点所在的局部社团。

近期,已经有研究者提出搜索局部社团的算法,例如 Aaron Clauset 提出局部模块度[8]概念,即通过最大化局部模块度为目标来搜索局部社团;James P. Bagrow 提出了搜索局部社团的算法[9],提出 shell 的概念,从局部开始进行广度搜

到稿日期:2009-08-26 返修日期:2009-10-09 本文受 863 国家高技术研究发展计划(2006AA01Z177),国家自然科学基金(60873027)资助。
李孔文(1982—),女,硕士生,主要研究方向为分布式计算与并行处理,E-mail:likongwen@gmail.com;顾 庆(1973—),男,教授,主要研究方向为分布式处理、软件工程等;张 尧(1984—),男,硕士生,主要研究方向为分布式计算与并行处理;陈道蓄(1949—),男,教授,博士生导师,主要研究方向为分布式计算与并行处理、软件工程等。

索，设定 ΔK 的阈值来确定局部社团的节点；以及解诒等提出了基于节点度优先的快速局部社团划分算法[11]。

本文提出一种局部社团划分的新算法，通过搜索邻居节点，选择符合条件的节点加入局部社团，最终获得待求节点所在的社团。局部社团对邻居节点的吸引力取决于社团对该节点的重要程度。一般地，如果局部社团的某个邻居节点主要通过该社团与其邻居节点进行连接，那么这个社团对该邻居节点具有重要影响，该邻居节点趋向于成为这个社团中的一员。节点的聚集系数就是表征某节点邻居之间连接的紧密程度，因此可以考察某局部社团存在前后其邻居节点聚集系数的变化情况，来确定社团和节点之间的关系。该算法也可用于全局社团划分，即利用该算法寻找到某个节点的社团后，从网络中社团外的任一节点开始寻找其社团，重复此过程，即可得到网络的全局社团结构。

“基于聚集系数的局部社团划分算法”能够利用网络自身的连接特点，在只需要确定邻居节点的连接情况下确认节点所在的局部社团。该算法可以对难以获得全局网络信息或者规模很大的网络，进行局部社团划分，也可以进行全局社团划分。

2 定义

关于网络中的社团结构的概念目前没有统一被认可的定义。较为常用的定义是“网络由社团构成，社团内部的节点之间连接紧密，社团之间的节点连接稀疏”，而“紧密”和“稀疏”并没有明确的标准。为了更加方便地探索网络中的社团结构，研究者试图提出一些定量化的社团定义，例如“强社团”和“弱社团”的定义。以下列出和本文算法相关的概念及定义：

1)强社团：满足下面条件的社团 ζ 称为强社团[4,5]：

$k_i^{in}(\zeta)>k_i^{out}(\zeta),\forall i\in\zeta$

式中，$k_i^{in}(\zeta)$ 表示节点 i 与 ζ 内部节点连接的度，$k_i^{out}(\zeta)$ 表示 i 和 ζ 外部节点连接的度。即在强社团 ζ 中，任何一个节点与 ζ 内部节点连接的度大于其与 ζ 外部节点连接的度。

2)聚集系数：分为节点的聚集系数和社团的聚集系数。

a)节点 i 的聚集系数 $c(i)$ 定义为：

$c(i)=E(i)/T(i)$

式中，设节点 i 的度是 k，$T(i)$ 表示节点 i 的 k 个邻居节点之间可能形成的最大连接数：$T(i)=k(k-1)/2$；$E(i)$ 表示节点 i 的邻居节点之间实际的连接边数。

聚集系数 $c(i)\in[0,1]$，当 $c(i)=1$ 时，表示节点 i 的所有邻居节点之间相互连接，i 和它的邻居节点之间形成了全局耦合网络，连接紧密。

b)社团 ζ 的聚集系数 c_ζ 为：

$$c_\zeta=\frac{\sum_{i=1}^{n}c(i)}{n}$$

式中，n 表示社团内节点的个数。

聚集系数 $c_\zeta\in[0,1]$，当 $c_\zeta=1$ 时，社团内所有节点的聚集系数均为 1，社团内的节点和社团外的邻居节点连接紧密。

3)邻居节点集

a)节点 i 的邻居节点集：

$N(i)=\{j|$节点 i 和节点 j 直接相连$\}$。

b)具有 n 个节点的社团 ζ 的邻居节点集：

$N(\zeta)=\bigcup_{i=1}^{n}N(i)$。

4)连接相关度 Δc

社团 ζ 的邻居节点 j 与社团 ζ 的连接相关度 $\Delta c(\zeta,j)$ 为：

$\Delta c(\zeta,j)=c_1(j)-c_2(j)$

式中，$c_1(j)$ 表示节点 j 的聚集系数，$c_2(j)$ 表示去除社团 ζ 的全部连接后其邻居节点 j 的聚集系数。

5)社团的内度

设社团 ζ 有 n 个节点，社团 ζ 的内度为：

$D^{in}(\zeta)=\sum_{i=1}^{n}k_i^{in}(\zeta)/2$

6)社团的外度

设社团 ζ 有 n 个节点，社团 ζ 的外度为：

$D^{out}(\zeta)=\sum_{i=1}^{n}k_i^{out}(\zeta)$。

3 基于聚集系数的局部社团划分算法

3.1 算法介绍

本文提出的算法从待求节点开始，首先将该节点加入结果社团，即得到一个局部社团，计算社团所有邻居节点与其连接相关度，将与社团连接相关度最大的节点加入社团，形成新的局部社团，继续计算，选择重要的邻居节点。通过不断添加与社团连接紧密的邻居节点来划分最终社团，当社团对其所有邻居节点不再重要，即所有邻居节点不通过该社团与自身邻居节点进行相连时，可以停止搜索，获得结果社团。

为了优化算法，加快算法的收敛速度，在确定社团时，可以根据定义 1)、定义 2)提出以下 4 项约定：

1)约定 1：若存在节点 i 满足 $k_i^{in}(\zeta)>k_i^{out}(\zeta)$，则 $i\in\zeta$。即，如果社团外某节点一半以上的连接与该社团相连，那么该节点也在这个社团中。

2)约定 2：若存在邻居节点 i 满足 $c(i)=1$，则 $i\in\zeta\wedge N(i)\subseteq\zeta$。即，如果社团某个邻居节点的聚集系数为 1 时，那么该邻居节点以及和它相连的所有节点也在社团。

3)约定 3：如果社团的聚集系数为 1，那么该社团的所有邻居节点都在该社团。

4)约点 4：在局部社团中加入某邻居节点的条件是：该节点的聚集系数高于社团的聚集系数，并且与社团的连接相关度为邻居节点中的最大非负值。当社团中不再有符合这样条件的邻居节点时，局部社团形成。

结合 4 项约定，基于聚集系数的局部社团划分算法的描述见图 1。

3.2 算法分析

本节就局部社团划分算法分析两个方面的问题：其一为算法时间复杂度，其二为社团合并策略。

考虑该算法在寻找局部社团过程中，加入一个邻居节点需要的时间复杂度是 $O(e\times d)$，其中 e 表示社团内连接的边数，d 表示邻居节点的平均度数。

快速 GN 算法的时间复杂度是 $O(M\times N)$，M 和 N 分别表示网络的连接数和节点数。而当整个网络规模很大时，对于划分具有 n 个节点的局部社团的问题，本文算法只需要网络的局部信息就可以进行划分，由于 $e\leqslant n(n-1)/2$，同时 $n\ll N$，与需要全局信息的传统划分算法相比，该算法的时间复杂度较低。

在该算法中，局部社团从初始状态开始，搜索社团的邻居节点，始终选择当前最符合条件的节点加入社团，而不考虑任何相对较差的候选节点，因此得到的局部社团往往是局部最

优而不是全局最优的。特别地，社团倾向于选择聚集系数较高的邻居节点，使得社团的聚集系数变大，难以再加入新的邻居节点，划分得到的局部社团规模较小。

```
Input    i : target node
Output   ς(i) : local community that contains the target node
Declare  count ← 0      //the insignificant nodes counter
         N(ς), c_ς, c_1(j), c_2(j), Δc(ς, j)   //definition
Begin
   Add i to ς              //ς ≡ ς(i)
   //Step1—Step3: convention 1, 2, 3
   for ∀j ∈ N(ς) do
     Add node j over half of whose neighbors are in ς to ς
     Update N(ς); goto Step1;
   endfor

   Compute c_ς ;
   if c_ς == 1 then
     ς ← ς ∪ N(ς)
     Update N(ς); goto Step1
   endif

   for ∀j ∈ N(ς) do
     Compute c_1(j)
     if c_1(j) == 1 then
          ς ← ς ∪ N(j)
          Update N(ς); goto Step1
     endif
   endfor

   //Step4: add the node which is the max connectivity relevance
to ς
   for ∀j ∈ N(ς) do
     if c_1(j) < c_ς then
          Δc(ς, j) ← −∞
     else
          Compute c_2(j)
          Δc(ς, j) ← c_1(j) − c_2(j)
     endif
   endfor
   max ← 0 // max is the value of maximum Δc
   pos ← 0 // pos records the node which is the maximum Δc
   for ∀j ∈ N(ς) do
     if Δc(ς, j) < 0 then
          count ← count + 1
     else if Δc(ς, j) ≥ max then
          max ← Δc(ς, j); pos ← j
     endif
   endfor
   if count < |N(ς)| then
     ς ← ς ∪ {pos}
     Update N(ς); goto Step1
   else if |ς| == 1 then
     for ∀k ∈ N(ς) do
          Use this algorithm to compute ς(k)
     endfor
     Find ς(k) such that most edges of ς(i) to ς(k)
     ς(i) ← ς(i) ∪ ς(k)
     Output ς(i)
   else Output ς(i)
   endif
end
```

图 1　基于聚集系数的局部社团划分算法

利用该算法划分全局社团结构时，会得到一些规模较小的社团，可以对划分的社团进行合并。利用定义 5）和定义 6），分别计算小规模社团的内度和外度，当社团内度小于社团外度时，要将该社团和其它连接紧密的社团进行合并，重复迭代直到所有社团的内度都大于社团的外度。

3.3　算法示例

在具有 18 个节点的网络中（见图 2），搜索节点 18 所在的局部社团。根据算法：从节点 18 开始，形成社团 $\zeta=\{18,16,15,17\}$，$N(\zeta)=\{13,15,16,17\}$，计算连接相关度（见表 1），其中连接相关度最大的是节点 16。将节点 16 加入社团，则 $\zeta=\{16,18\}$，$N(\zeta)=\{13,14,15,17\}$。节点 15 和节点 17 超过一半的连接与 ζ 中的节点相连，将节点 15 和节点 17 加入社团，$\zeta=\{18,16,15,17\}$，$N(\zeta)=\{13,14\}$。节点 13 和节点 14 超过一半的连接在社团 ζ 中，将这两个节点加入社团 ζ，$\zeta=\{18,16,15,17,14,13\}$，$N(\zeta)=\{10\}$。此时，社团 ζ 的聚集系数 $c_\zeta=0.5444$，节点 10 的聚集系数 $c(10)=0.4$，节点 10 不能加入社团 ζ 中，局部社团形成，即寻找到节点 18 所在的社团（见图 3）。

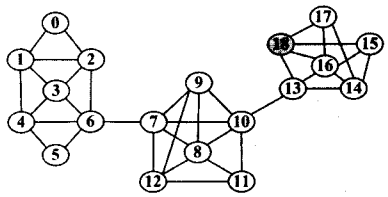

图 2　具有 18 个节点的网络，寻找节点 18 的社团

表 1　$\zeta=\{18\}$ 时的连接相关度计算表

N(ζ)	ζ_ζ	$c_1(j)$	$c_2(j)$	$\Delta c(\zeta,j)$
节点 13	0.5	0.3333<0.5	\	\
节点 15	0.5	0.6667>0.5	1	−0.3333
节点 16	0.5	0.6>0.5	0.5	0.1
节点 17	0.5	0.6667>0.5	1	−0.3333

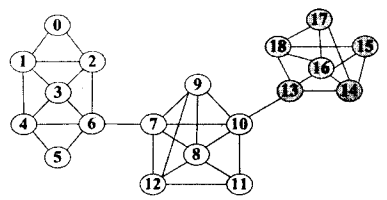

图 3　节点 18 所在的局部社团

4　算法应用及分析

4.1　“Zachary 空手道俱乐部网络”中社团划分的应用

“Zachary 空手道俱乐部网络”[10]是用来判断社团划分效果的常用实验网络。这个网络是在 20 世纪 70 年代初，由 Wayne Zachary 通过观察美国大学空手道俱乐部成员间的人际关系，并依据俱乐部成员间平时的交往状况建立的一个网络。这个网络包含 34 个节点，表示俱乐部成员；包含 78 条边，表示成员之间的人际关系。由于突发的原因，俱乐部管理者与俱乐部主要教师之间针对是否提高收费这一问题产生了激烈的争论，并最终导致俱乐部分裂成两部分。图 4 为空手道俱乐部网，其中方形顶点代表归于俱乐部管理者（1 号顶点）的成员，圆形顶点代表归于俱乐部主要教师（33 号顶点）的成员。

将本文提出的局部社团划分算法应用到空手道俱乐部网进行社团划分，从不同的节点开始搜索获得的社团结构基本一致，将网络划分为 5 个社团。从节点 1 出发，得到的网络社团结构见图 4，将实际网络中存在的两个社团分别细化成了 2 个、3 个社团。本文算法对 Zachary 网络的划分保持了实际网络中的成员属性，但是作出了更细层次的划分。

对算法的改进策略也可以用于实验结果的进一步分析。通过检测已划分的社团之间的连接频度，可以将连接较为紧密的社团进行合并，形成新的社团。例如在上述 Zachary 空

手道俱乐部网络中，对于已划分的 5 个社团 A,B,C,D,E，社团 A 的内度 $D^{in}(A)=3$，社团 A 的外度是 $D^{out}(A)=4$，并且 A 的邻居社团只有社团 B，将 A,B 两个社团进行合并，合并后的社团为 B_{new}，它的内度和外度分别为 $D^{in}(B_{new})=34$，$D^{out}(B_{new})=7$，这个合并后的社团不需要检测其邻居社团来进行合并，也形成了与实际网络一致的划分结果。

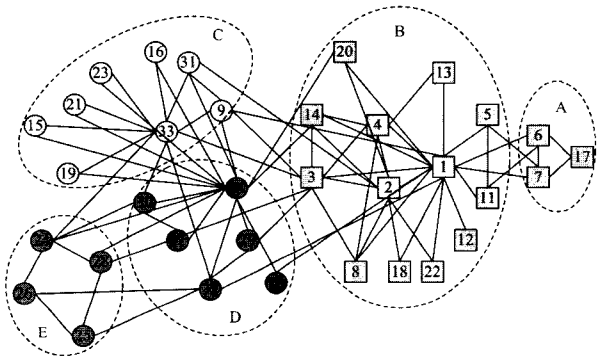

图 4　Zachary 空手道俱乐部网络，方形顶点代表实际网络中归于俱乐部管理者(1 号顶点)的成员，圆形顶点代表归于俱乐部主要教师(33 号顶点)的成员。相同颜色的节点代表由本文算法划分在一个社团的节点，共划分出 A,B,C,D,E 5 个社团

4.2　JSCG 网络中社团划分的应用

JSCG[12] 是在 SCG(Software Collaboration Graph)的基础上建立的符合 Java 语言的网络图结构，该图的节点是类、接口或者枚举类，边表示类、接口或者枚举类之间的相互关系，相互关系主要指继承关系、实现关系和聚集关系。文献[12]中介绍了构建 JSCG 网络的详细步骤。图 5 是一个构建 JSCG 网络的例子。

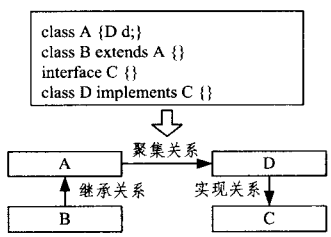

图 5　一个简单的 Java SCG 的构建实例

JDK 是 Java 开发工具包(Java Development Kit)的缩写。根据文献[12]构建 JSCG 的方法，本文对 JDK1.6.0_11 中常用的几个包的代码结构进行分析并建立相应的 JSCG 图。Java 编译器是为每个类生成一个对应的字节码文件，因此首先分析 JDK 中的字节码文件，依次读入需要分析的类字节码文件，将一个字节码文件中读入的类、接口或者枚举类作为一个节点，而将它们之间的继承关系、实现关系和聚集关系作为连边的条件，构建 JSCG 网络图。

在已构建的 JSCG 网络中，利用本文算法分别对不同规模的包进行社团划分，统计不同包形成的社团个数和社团规模，实验数据见表 2 和图 6。其中，孤立节点是指 JSCG 图中度为 0 的节点，即在包中没有和其它包中的类、接口或者枚举类有任何关系的类。

表 2　JDK 中部分包的社团划分结果

Pacakge Name	节点数	边数	孤立节点数	社团个数
Java. nio	150	146	32	46
Java. lang	229	284	7	21
Java. awt	556	784	76	136
Java. util	644	1027	84	120
Java	2406	6306	16	321
Javax	3327	4599	648	681

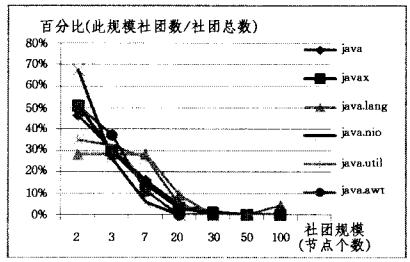

图 6　JDK 中部分包的社团划分规模分布图

JDK 是一种用于构建在 Java 平台上发布的应用程序、applet 和组件的开发环境。它是一切 Java 应用程序的基础，所有的 Java 应用程序都是构建在它之上的。

在 JDK 设计中，每个包中节点之间连边较少，基于聚集系数的局部社团划分算法将 JDK 中不同包的 JSCG 划分成多个社团，划分出的社团规模都比较小，在每个包中也只有 1 到 2 个规模较大的社团。这是由于 JDK 作为 Java 软件开发的基础类库，主要提供了底层功能的接口(或抽象类)和少量基本实现，单个功能模块所涉及的类数量相对较少。本算法严格的社团界定标准导致只有各个功能模块内的类被划分在一个社团，因此得出了大量小规模社团的划分结果。

例如，在 java. nio 包中有 67%的社团中仅有 2 个节点。java. nio 的全称是 java new I/O，这个包主要定义了 Buffer 及其子类。Buffer 定义了一个线性存放基本数据类型的容器接口。对于除 boolean 以外的其他基本类型，都有一个相应的 Buffer 子类，而对每个基本类型的 Buffer 子类，nio 包都提供了基于 ByteBuffer(和直接存取)的一个基础实现和只读实现，以 CharBuffer 为例，其子类体系结构见图 7。其扁平的结构是与本方法所划分出的以节点数为 2 的社团为主的特点相吻合的。

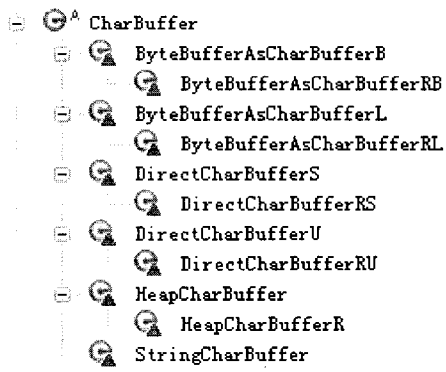

图 7　CharBuffer 子类体系结构图

结束语　对大型复杂网络进行社团划分已经成为复杂网络中的一项研究重点，在实际大型复杂网络中获得全局网络信息非常耗时甚至很困难，而在某些情况下仅需要了解某节点所在的局部社团信息。基于这些实际需要的考虑，本文提出一种基于节点聚集系数的局部社团划分算法，从单个节点出发搜索该节点所在的社团。这种算法从局部开始，不需要

(下转第 53 页)

的吞吐率比值为 0.761。可以看出，TFRC 较好地保持了对 TCP 业务流的友好性。在应用 TFRC 算法时，TFRC 流的平均延时抖动为 0.0110ms；在应用 TFRC-CJUTD 算法时，TFRC-CJUTD 流的平均延时抖动为 0.0106ms；在应用 TFRC-JI 算法时，TFRC-JI 流的平均延时抖动为 0.0099ms，相对 TFRC 以及 TFRC-CJUTD 分别下降了 10%和 6.6%。以上数据表明，和已有的 TFRC 及改进算法比较，TFRC-JI 在保持对 TCP 业务友好性的同时实现了链路的高效使用，并降低了传输时延抖动，从而较好地适应多协议共存的无线网络实时业务传输。

结束语 传统基于 UDP 的应用尤其是多媒体应用对带宽资源争用严重，这使得它在与 TCP 流共存的网络中表现出了对 TCP 极大的攻击性。本文通过深入分析典型的 TCP 友好拥塞控制协议 TFRC 及其在无线环境下的不足，提出了一种改进的 TFRC-JI 机制，它通过传输延时抖动有效区分无线链路的拥塞和误码，并以此反馈至发送端，实现不同的速率控制机制。仿真实验验证了 TFRC-JI 在保持对 TCP 业务友好性的同时实现了链路的高效使用，并降低了实时业务流的传输时延抖动，从而较好地适应多协议共存的无线网络实时业务传输。

参 考 文 献

[1] 刘绍辉. Internet 拥塞控制相关算法研究及仿真分析[D]. 成都：西南交通大学，2005

[2] Widmer J, Denda R, Mauve M. A Survey on TCP-friendly Congestion Control[J]. IEEE Network Magazine, 2001, 15(3): 28-37

[3] Yang Y, Lam S. General AIMD Congestion Control[C]// Proceedings of ICNP 2000. Osaka, Japan, Nov. 2000

[4] Bansal D, Balakrishnan H. Binomial Congestion Control Algorithms[C]// Proceeding of IEEE INFOCOM 2001. Anchorage, Alaska, Apr. 2001: 631-640

[5] Rejiaie R, Handly M, Estrin D. RAP: An End-to-End Rate-Based Congestion Control Mechanism for Realtime Streams in the Internet[C]// Proceedings of IEEE INFOCOM99. Mar. 1999

[6] Sisalem D, Wolisz A. LDA+: TCP-friendly Adaptation Scheme for Multimedia Communication [C] // Proceedings of NOSSDAV'98. Cambridge, England, July 1998

[7] Rhee I, Ozdemir V, Yi Y. TEAR: TCP emulation at receivers-flow control for multimediastreaming[R]. Dept. of Comp, Sci, NCSU, Apr. 2000

[8] Handley M, Floyd S, Padhye J, et al. TCP Friendly Rate Control (TFRC)[S]. Protocol Specification, RFC3448. IETF, 2003

[9] 吴东. TCP 友好拥塞控制研究[D]. 南宁：广西师范大学，2006

[10] 甘泉，薛质. 控制端到端传输延迟抖动的改进 TFRC 算法[J]. 计算机工程，2008，34(10)：105-107

[11] Wen P, Cao J, Li Y. Design of High- performance Networked Real time Control Systems [J]. Control Theory & Applications, 2007, 1(5): 1329-1335

[12] Zhou B, Fu C P, Chiu D M, et al. A Simple Throughput Model for TCP Veno[C]// IEEE ICC 06. June 2006

(上接第 49 页)

考虑全局信息，在时间复杂度上有一定优势。同时，这种局部社团划分算法也可以应用于全局网络的社团结构的搜索，并且不需要事先了解社团规模或者全局社团个数。把这种算法应用在 Zachary 俱乐部网络进行实验，取得了较好的划分效果。该算法对 JDK 的 JSCG 网络图的划分结果也证实了 JDK 提供的功能模块的特点。但是该算法在选择节点时都是选择局部最优值，可能最终产生的解不是全局最优解，下一步工作将考虑结合模拟退火算法对其进行改进。

在软件开发设计过程中，需要将软件划分成不同模块进行开发。在 Java 软件开发设计中，当确定了类、接口等之间的相互关系后，可以对生成的 JSCG 图利用本算法进行社团划分从而形成开发的模块，以便于软件设计和开发。下一步工作也将应用此算法对其它软件进行社团划分和分析。

参 考 文 献

[1] Watts D J, Strogatz S H. Collective dynamics of 'small world' networks[J]. Nature, 1998, 393: 440-442

[2] Barabási A L, Albert R. Emergence of scaling in random networks[J]. Science, 1999, 286: 509-512

[3] Newman M E J. Detecting community structure in networks[J]. J. Eur Phys J B, 2004, 38: 321-330

[4] Girvan M, Newman M E J. Community structure in social and biological networks[J]. J. Proc. Natl. Acad. Sci. USA, 2002, 99: 7821-7826

[5] Radicchi F, Castellano C, Cecconi F F, et al. Defining and identifying communities in networks[J]. Proc Natl Acad Sci USA, 2004, 101: 2658-2663

[6] Newman M E J, Girvan M. Finding and evaluating community structure in networks[J]. J, Phys Rev E, 2004, 69: 026113

[7] Newman M E J. Fast algorithm for detecting community structure in networks[J]. Phys Rev E, 2004, 69: 066133

[8] Clauset A. Finding local community structure in networks[J]. J, Phys Rev E, 2005, 72: 026132

[9] Bagrew J P, Bollt E M. A Local Method for Detecting Communities[J]. J, Phys Rev E, 2005, 72: 046108

[10] Girvan M, Newman M E J. Community structure in social and biological networks[J]. J, Proc Natl Acad Sci USA, 2002, 99: 7821-7826

[11] 解冶，汪小帆. 复杂网络的一种快速局部社团划分算法[J]. 计算机仿真，2007，24(11)：82-85

[12] 陈焘，顾庆，王树森，等. 基于复杂网络的 JDK 代码结构演化研究[J]. 电子学报，2007，35：118-123

[13] Maqbool O, Babri H A. Hierarchical Clustering for Software Architecture Recovery[J]. IEEE Trans. on Software Eng, 2007, 33 (11)

[14] 杨博，刘大有，Liu Jiming，等. 复杂网络聚类方法[J]. 软件学报，2009，20：54-66