文章编号:1001 - 9081(2008)01 - 0162 - 03

P2P 网络中的数据挖掘

刘天鹏,周 娅
(桂林电子科技大学 计算机与控制学院,广西 桂林 541004)
(liutp@163.com)

摘 要:在分析了现有分布式数据挖掘算法的运行机制和 P2P 技术具有无中心、不同步等特点的基础上,通过扩展经典 K-mean 算法的迭代过程,设计了一种能够用于 P2P 网络的分布式数据挖掘算法。该算法只需要在直接相连的节点间传递数据,并且能使每个节点上的数据按照全局聚类的结果聚合。最后用模拟实验验证了该算法的有效性。

关键词:K-mean 算法;分布式数据挖掘;对等网;聚类
中图分类号: TP311 **文献标志码**:A

Data mining in P2P networks

LIU Tian-peng, ZHOU Ya
(*School of Computer & Control, Guilin University of Electronic Technology, Guilin Guangxi* 541004, *China*)

Abstract: To analyze both the operational mechanism of current distributed data mining and the characteristics of the P2P technology: non-centralized peer and asynchronism, by extending the iterative process of classical K-mean algorithm, a distributed data mining algorithm was designed in this paper to implement k-mean thinking in a P2P networks. This algorithm exchanges information only between directly connected nodes, and can cluster local data on each peer in a global view. Finally, simulation experiments show that the algorithm is effective and accurate.

Key words: K-mean; distributed data mining; P2P; clustering

由于 P2P 技术的流行,使 P2P 结构的网络上储存了大量的数据。P2P 网络中的节点同时充当这些资源的拥有者和需求者。如果对这样的一个含有大量数据的系统应用数据挖掘技术一定会得到许多有用的信息。目前在 P2P 网络中应用的数据挖掘技术主要是为了构造更为合理的拓扑结构,而不是为了挖掘存储在 P2P 网络中的信息。传统的分布式数据挖掘多是基于 C/S 结构,需要将数据集中到中心站点上,而多处理机的并行数据挖掘算法大多需要处理机之间的消息广播,它们并不能适用于 P2P 这种没有中心、结构松散,节点处理能力参差不齐的网络。本文通过聚类算法研究如何在 P2P 网络中挖掘数据。

1 相关工作

1.1 典型的 P2P 网络拓扑

根据拓扑结构的关系可以将 P2P 技术分为 4 种形式:中心化拓扑,全分布式非结构化拓扑,全分布式结构化拓扑和半分布式拓扑[1]。中心化拓扑依赖中心化的目录系统,发现算法灵活高效并能够实现复杂查询。全分布非结构化网络采用了随机图的组织方式,对网络的动态变化有较好的容错能力。全分布式结构化拓扑采用 DHT 技术,将节点按照一定的规律连接起来并将数据分散在节点中。半分布式拓扑是介于全分布非结构化拓扑和中心化拓扑之间的一种拓扑结构,它没有中心节点,但是有一组比较稳定的节点充当的超级节点,它兼有两者的有点,应用最为广泛。

1.2 现有的分布式聚类算法

虽然 P2P 的资源共享系统已经十分流行(如 BT,eMule 等文件共享系统),但是在 P2P 网络中实现数据挖掘的研究还比较少,以下列举了一些并行聚类和分布式聚类的研究。文献[2,3]提出了一个并行 K-mean 算法,这个算法运行在一个多处理器的环境中,每一个处理器都会广播自己的聚类质心,当每个处理器接受到其他全部处理器的消息后就可以求出全局聚类质心。文献[4,5]提出了一种用于分布式系统的层次聚类方法,它是将数据先在各个分站点上完成聚类,然后各个分站点再将自己聚类的统计信息发送到中心站点,最后由中心站点完成全局聚类。文献[6]提出了一种基于密度的分布式聚类方法,与前面的文献类似,它先将数据分发到各个站点,先由各个分站点完成本地聚类,最后将数据发回中心站点完成全局聚类,但是分站点传给中心站点的信息有所不同。文献[7]提出了一种用于传感器网络的异步 K-mean 聚类算法。文献[8]提出了一种在 CAN 网络上实现的基于密度的异步层次聚类算法,并且要求被聚类数据的空间与 CAN 的路由空间一致;它首先是让每个节点对本地的数据进行聚类,然后按照 CAN 划分路由空间的逆过程合并聚类,最后得到全局的聚类结果。

2 P2P 对数据挖掘的新要求

由于 P2P 网络的特殊性,在这样的环境中实现数据挖掘与经典数据挖掘算法和 C/S 模式的分布式数据挖掘有着很大的不同。在 P2P 网络环境中有许多因素需要考虑,比如节点的计算能力,内存大小,带宽大小等,有时还要考虑每个节点由于通信的电力消耗。因此为了在 P2P 网络中实现数据挖掘,要求数据挖掘算法具有以下几个特点:

1)完全分布式:没有中心节点是 P2P 网络的重要特点,所以 P2P 中的数据挖掘算法需要使数据在网络上计算而不

收稿日期:2007 - 07 - 13;**修回日期**:2007 - 09 - 12。 **基金项目**:广西自然科学基金资助项目(桂科字 0447091)。

作者简介:刘天鹏(1982 -),男,安徽蚌埠人,硕士研究生,主要研究方向:数据挖掘、对等网计算; 周娅(1966 -),女,湖北松滋人,教授,主要研究方向:数据库安全、数据挖掘、计算机网络。

是在某个中心节点上计算。

2)可扩展性:可扩展性无疑是 P2P 网络能够迅速流行的重要原因,中心节点的瓶颈和广播消息泛滥是制约可扩展性的主要方面。

3)流量控制:在 P2P 应用程序中,数据挖掘算法的网络开销不应该成为网络的主要开销,所以 P2P 网络中的数据挖掘算法应该是"轻量级"的,它应该使用最少的网络开销。

4)非同步性:在一个巨大而且结构松散的 P2P 网络系统中,要想使所有节点按照统一的步调执行某个算法是不可能的,所以在 P2P 网络中的数据挖掘算法不能是全局同步的。

综上所述,在 P2P 网络中的数据挖掘算法应该是不依赖中心节点处理数据,不依赖洪泛传播消息,并且能够得到近似集中式数据挖掘算法的结果的"轻量级"算法。

同时对于不同结构的 P2P 网络实现聚类算法也有不同的考虑,例如对于 Gnutella 和 Ad Hoc 网络就比较适合于依靠"中心"区分的 K-mean 算法,而 CAN 网络更适合主要关注"聚类边缘"的 DBSCAN 和 K-NN 等算法。本文研究的内容是设计一种能够用于 Ad Hoc 型网络的 K-mean 算法。

3　P2P 网络中的 K-mean 算法

在给出 KIPN(K-mean In Peer-to-peer Networks)算法的详细描述前先给出一些定义并且对将要用的符号进行说明:

定义 1　为了减少网络负载,节点间不可能交换全部的聚类数据,而是用包括聚类的大小,分布等统计特征来代替,这个统计信息叫作聚类的质心。

定义 2　在 KIPN 的聚类质心中,描述质心位置的坐标称为聚类的中心,类似经典 K-mean 算法中的数据中心。

N_i 表示 P2P 网络中的节点,X_i 表示这个节点上存储的局部数据。全局的数据集合为 $X = \bigcup_{i=1}^{n} X_i$,两个数据元素 x_i 和 x_j 间的距离为 $|x_i - x_j|$,用 $Neighbor_{(i)}$ 表示与 N_i 直接相连的节点集合。用一个四元组描述一个聚类的质心 $C_i = \{PeerID, ClusterID, Center, Count\}$,其中 *PeerID* 是产生这个聚类的节点的编号;*ClusterID* 是聚类的编号,它是一个全局唯一的标识符;*Center* 表示聚类质心的中心;*Count* 表示聚类的权重(一般说是聚类包含的数据元素的个数)。

3.1　算法的框架描述

与经典的 K-mean 算法类似,KIPN 算法也是一个逐步迭代的过程,步骤如下:在开始迭代前,节点 N_i 首先对自己的本地数据做一次 K-mean 算法得到初始聚类 $\{C_i \mid 1 \leqslant i \leqslant K\}$;完成初始聚类的节点 N_i 向它的所有邻居节点 $Neighbor_{(i)}$ 发送聚类请求消息 $Request(i)$。当邻居节点 $N_k(k \in Neighbor_{(i)})$ 接收到消息 $Request(i)$ 后就将自己聚类的质心集合 $Response(k)$ 发送给节点 N_i。当节点 N_i 接收到它的所有邻居发送来的 *Response* 消息后就使用这些信息更新自己的本地聚类质心;节点 N_i 上的算法终止条件为 $\max\{|Center'_j - Center_j|\} \leqslant e$(其中 $Center'_j$ 为更新后的聚类中心,e 为预设的最大误差),否则继续向邻居节点发送 $Request(i)$ 消息,继续迭代过程。

算法 1　KIPN 框架

输入:节点局部数据 N_i 和迭代过程中接收邻居节点的 $Response(k)$ 消息

输出:生成的聚类质心集合 *Center* 和迭代过程中向邻居节点发出的 $Request(i)$ 消息

```
Classical_K_mean (Ni);          //用经典 K-mean 聚类本地数据
while not Terminated do
begin
  foreach Neighbors
    Send_Message (Request (i));
                    //向邻居节点发送请求 Request(i)消息
  foreach Neighbors
    Receive_Message (Response (k));
                    //等待邻居节点的回应 Response(k)消息
  Update_Local_Clusters (Ni);
                    //根据邻居节点的回应更新本地聚类质心
  if (max {|Centerj' - Centerj|} > e)
            //计算更新后聚类质心的变化,判断算法是否中止
    Terminate = false;
  else
    Terminate = true;
end;
```

KIPN 算法的框架描述了它的行为和执行步骤,其核心部分是更新本地聚类质心。这个算法在各个参加运算的节点上运行,算法的复杂度可以分为两个部分:更新聚类质心和向邻居节点发送消息。更新聚类质心的复杂度主要依赖数据在网络中的分布情况,根据小世界模型[9],每个节点的更新算法的时间复杂度为常数(在仿真实验中得到验证);发送消息数主要依赖网络中与之直接相连的节点数,每个更新过程发送一次消息。

3.2　更新本地聚类质心

KIPN 算法的核心是节点如何更新本地聚类质心。当一个节点 N_i 收到所有邻居的 $Response(k)$ 消息后就要对自己的本地聚类信息进行更新。在经典的 K-mean 算法中,聚类中心使用节点坐标和的平均值来表示,如式(1)所示:

$$Center'_j = \frac{\sum_{i=0}^{n} x_i}{n}; x_i \in C_j, n = |C_j| \tag{1}$$

但是在 KIPN 算法中参与更新计算的不是具体的数据元素而是这些数据元素在别的站点上的聚类的统计信息,所以更新时不仅要更新聚类的中心坐标还要更新聚类的大小。在这里假设每个聚类内部的数据是均匀分布的。

定义 3　某个聚类质心相对于全部聚类中心而属于这个中心的比重称为聚类质心对中心的隶属度,计算公式如下:

$$m_{i,j} = \frac{\frac{1}{|C_i.Center - Center_j|}}{\sum_{i=0}^{k} \frac{1}{|C_i.Center - Center_i|}} \tag{2}$$

利用上面定义的隶属度,将经典 K-mean 算法的中心更新公式扩展为各个聚类按权重计算新聚类中心的形式,更新公式表示如下(其中 N 表示这个节点上的所有聚类,包括接收到的邻居节点的聚类):

$$Center'_j = \frac{\sum_{i=0}^{N} \left(C_i.Center \times C_i.Count * m_{i,j}\right)}{\sum_{i=0}^{N} \left(C_i.Count \times m_{i,j}\right)} \tag{3}$$

新生成聚类的大小为各个聚类大小的隶属度加权和,更新公式如下:

$$Count'_j = \frac{\sum_{i=0}^{N} \left(C_i.Count \times m_{i,j}\right)}{\sum_{i=0}^{N} m_{i,j}} \tag{4}$$

在计算某个聚类质心的归属的时候,不是简单的将聚类质心归于某个中心,而是通过计算聚类质心对各个中心的隶

属度,将聚类质心按一定的比重分配给各个聚类中心。最后通过式(3)和(4)更新聚类中心和聚类大小。

3.3 聚类结果的表示

与经典的 K-mean 算法的设计目的不同,P2P 网络系统中很难也没有必要得到一个全局的聚类结果,KIPN 算法的目标是得到在全局的观点下各个节点的局部聚类结果。下面通过一个例子说明什么是“全局的观点下各个节点的局部聚类结果”。

考虑这样一组一维空间的点并将其依次按坐标值从小到大编号,如图 1 所示,图中圆点的距离表示数据间的相似性。可以用 K-mean($K=2$)算法将其聚类成两类{(1,2,3,4,5),(6,7,8,9,10,11,12)}。

①②③④⑤ ⑥⑦⑧⑨ ⑩⑪⑫

图 1 聚类数据

将这些数据分散到两个节点上 $P_1(1,2,3,4,5,6,7)$ 和 $P_2(8,9,10,11,12)$,如果是单纯在每个节点上做 $K=2$ 的 K-mean 聚类,得到的结果会是 P_1{(1,2,3,4,5),(6,7)} 和 P_2{(8,9),(10,11,12)},对于 P_1 聚类的结果和全局的结果是一样的,所以 P_1 的结果是“全局的观点下各个节点的局部聚类结果”,但是对于 P_2 聚类的结果就不符合全局聚类结果,这是因为 K-mean 算法一定会把数据分为 k 类,而 P_2 上的数据在全局观点上只有 1 类,所以 P_2 上的聚类结果一定是不正确的。

定义 4 对于某个局部的数据集合中的两个数据元素 d_1 和 d_2,他们是属于同一个分类的当且仅当他们在全局聚类结果中也是属于一类的,这样的局部聚类结果叫作“全局的观点下各个节点的局部聚类结果”。

4 实验分析

为了验证 KIPN 算法的有效性,我们进行了一系列仿真实验。利用 Windows 的多线程机制,使用每个线程代表一个节点,用线程间的通信模拟节点间的 socket 通信。实验数据采用人工生成的二维数据集,共有 65000 个元素,包含了密度不同并且彼此间可能存在公共区域的 4 个聚类,以及一些随机噪音(如图 2 所示),并将这些数据随机的分配到各个节点上。

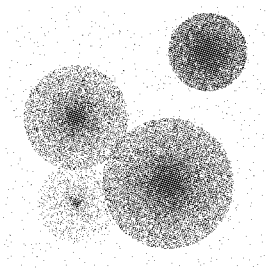


图 2 实验数据分布

实验分析主要从聚类的效率和准确性两方面来考查。在具体介绍实验内容前先给出准确性参数的一些说明。我们使用两个参数 α 和 β 考查算法的准确性,分别代表两种不同的错误,$1-\alpha$ 代表在节点的局部聚类中属于一类而在全局观点下不是一类的百分比,$1-\beta$ 表示在局部聚类时没有分到一类而在全局观点中属于一类的百分比。聚类的效率主要是算法给网络的负载,我们使用节点间的发送的消息量来表示。

实验中我们假设网络中任意两个节点间以 0.05 的概率存在直接连接,分别在含有 50,100,150,200,250,300,350,400,450,500 等不同节点数量的情况下模拟试验,KIPN 算法的通信量和聚类准确度如图 3 和图 4 所示。

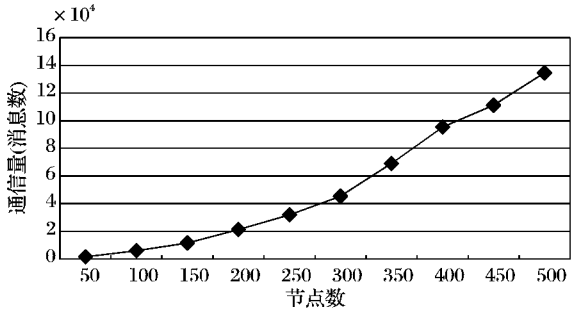


图 3 聚类算法通信量

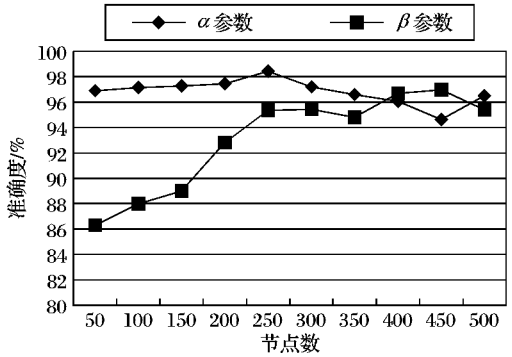


图 4 聚类算法准确度

如图所示,α 参数在节点数量增加的同时保持了相对的稳定,而 β 参数随节点数量的增加有所增加。这主要是由于在 KIPN 算法的迭代过程中采用的是合并聚类质心的策略,而且实验采用随机等概率的向每个节点分配数据,所以当节点数量较多的时候每个节点上产生的原始聚类就比较的小,因而会影响最终的结果。

KIPN 算法的单个节点平均通信量与节点数量的增长关系如图 5 所示。

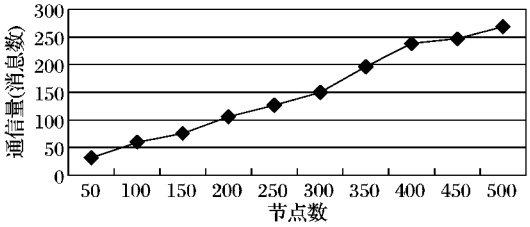


图 5 单个节点通信量

通过实验数据的分析可以发现 KIPN 算法的单个节点平均通信量基本与节点数量的增长成线性关系,由于节点的邻居数与节点个数成正比,根据算法 1 所示的框架可以得出 KIPN 算法中单个节点的迭代次数基本是常数。这也符合由小世界模型得出的每个节点的更新算法时间复杂度为常数的结论。

5 结语

综上所述,本文通过对现有分布式数据挖掘算法的研究,结合 P2P 网络的特点,在经典 K-mean 算法的思想基础上提出了 P2P 环境下的数据聚类算法“KIPN 算法”,并通过模拟实验验证了 KIPN 算法的有效性。

本文的下一步工作包括在此基础上进一步分析实验结果

(下转第 170 页)

上提出了一种自适应修改搜索范围的改进算法：

第 1 步：通过 Douglas-Peucker 算法快速计算出一条参考压缩路径 $P = \{p(0), p(1), \cdots, p(H)\}$，然后根据此参考路径 P 进行后续步骤的迭代求解，并且前一次的结果作为下一次迭代的参考路径。

第 2 步：构造最小误差的搜索空间 Ω'，加快搜索速度。因为在第 1 步中已经计算出了一条参考路径，可以通过这一参考路径构造一个搜索条带，从而减少搜索空间。因为路径中各点的拐角程度不同，即压缩度不同，所以搜索条带的宽度 W 可根据压缩度自适应变化。改进的搜索空间 Ω' 的边界函数如下：

$$L(h) = \begin{cases} \max\{h+1, h+(p(h)-p(h-1)) \times H/Nb\}, & h = 1,2,\cdots,H \\ h+1, & h = 0 \end{cases}$$

$$R(h) = \begin{cases} \min\{Nb, h+2\times(p(h)-p(h-1)) \times H/Nb\}, & h = 0,1,\cdots,H-1 \\ Nb, & h = H \end{cases}$$

$$B(nb) = \begin{cases} 0, & nb = 0 \\ h, & nb = R(h-1)+1,\cdots,R(h) \end{cases}$$

$$T(nb) = \begin{cases} \min\{M, h+2\times(p(h)-p(h-1)) \times H/Nb-1\}, & nb = L(h),\cdots,L(h+1)-1 \\ H, & nb = Nb \end{cases}$$

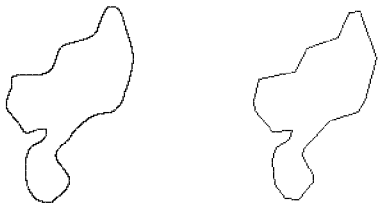
第 3 步：在搜索空间 Ω' 中运用动态规划算法求解。

这个改进算法的主要计算时间可以分为两部分：计算压缩误差和搜索最小压缩误差值。因为对于每一个节点 f_{nb} $(nb = 1,2,\cdots,Nb)$，都要计算与其他节点的误差，所以误差计算部分的时间复杂度为 $O(WNb^2/H)$；因为共有 WH 个状态空间点需要计算 $D(nb,h)$，所以最小误差搜索部分的时间复杂度为 $O(WNb/H)$。显然，此算法的时间复杂度为 $O(Nb^2W^2/H)$。

3 实验结果与分析

实验采用一条等高线作为原始数据，取压缩率 $\eta = 95\%$，图 2 为实验结果。

图示等高线中，节点数为 $Nb = 2\,658$，$Ne = Nb \times (1-\eta) = 133$，原始动态规划算法的计算时间为 31 s，本文提出的改进算法的计算时间为 4 s，显然在效率上有很大的提高，同时其最小误差与原始算法基本相同，比普通的 Douglas-Peucker 算法要小得多。



(a) 压缩前 (b) 压缩后

图 2 基于动态规划算法的矢量数据压缩改进算法结果

4 结语

对于矢量数据而言，由于其复杂性与数据格式多样性，对其进行压缩有一定的难度。本文针对线状图形要素，以曲线压缩后节点数较少和误差较小为优化目标，研究基于动态规划算法的矢量数据压缩模型和方法，提出了一种自适应构造最优搜索条带宽度的方法。众多实验表明，计算时间大大缩短，压缩误差显著减少，取得了较好的压缩效果。此算法可以提高移动设备中矢量数据的存储能力和矢量数据在无线网络上的传输速度。本文主要是研究单空间实体的优化压缩方法，而没有考虑多实体间的整体优化关系，下一步应在这一方面进行深入研究，逐步完善矢量数据优化压缩方法。

参考文献：

[1] PEREZ J C, VIDAL E. Optimum polygonal approximation of digitized curves[J]. Pattern Recognition Letters, 1994, 15(2): 743 - 750.

[2] KOLESNIKOV A, FRANTI P. Reduced-search dynamic programming for approximation of polygonal curves[J]. Pattern Recognition Letters, 2003, 24(14): 2243 - 2254.

[3] KOLESNIKOV A, FRANTI P. Data reduction of large vector graphics[J]. Pattern Recognition, 2005, 38(3): 381 - 394.

[4] 杨建宇. 基于组件的分布式地理信息服务研究[D]. 北京：中国科学院遥感应用研究所, 2005.

[5] 杨建宇, 杨崇俊, 明冬萍, 等. WebGIS 系统中矢量数据的压缩与化简方法综述[J]. 计算机工程与应用, 2004, 40(32): 36 - 38.

[6] 陈飞翔. 移动空间信息服务关键技术研究[D]. 北京：中国科学院遥感应用研究所, 2006.

[7] 翟战强, 管华, 王双亭. 一种快速空间矢量数据压缩方法[J]. 计算机工程, 2003, 29(2): 94 - 95.

(上接第 164 页)

的基础上改进 KIPN 算法的效率，同时研究将更多的数据挖掘算法在 P2P 环境下实现的方法。

参考文献：

[1] 罗杰文. Peer to Peer (P2P) 综述[EB/OL]. [2005 - 11 - 03]. http://www.intsci.ac.cn/users/luojw/papers/p2p.htm.

[2] DHILLON I S, MODHA D S. A data-clustering algorithm on distributed memory multiprocessor [C]// Large-Scale Parallel Data Mining, Workshop on Large-Scale Parallel KDD Systems. London: Springer-Verlag, 1999.

[3] FORMAN G, ZHANG B. Distributed data clustering can be efficient and exact[J]. ACM SIGKDD Explorations Newsletter, 2000, 2(2): 34 - 38.

[4] JOHNSON E L, KARGUPTA H. Collective, hierarchical clustering from distributed, heterogeneous data[C]// Large-Scalle Parallel KDD System. London: Springer-Verlag, 1999.

[5] SAMATOVA N F, OSTROUCHOV G, GEIST A, *et al.* RACHET: An efficient cover-based merging of clustering hierarchies from distributed datasets[J]. Distributed and Parallel Databases, 2002, 11(2): 157 - 180.

[6] JANUZAJ E, KRIEGEL H P, PFEIFLE M. DBDC: Density based distributed clustering[C]// Proceedings of International Conference on Extending Database Technology (EDBT). Heraklion: [s. n.], 2004: 88 - 105.

[7] BANDYOPADHYAY S, GIANELLA C, MAULIK U, *et al.* Clustering distributed data streams in peer-to-peer environment[J]. Information Science Journal, 2005, 176(14): 1952 - 1985.

[8] MEI L, GUANLING L, CHIEN L W. PENS: An algorithm for density-based clustering in peer-to-peer systems[C]// Proceedings of the 1st international conference on Scalable information systems. New York: ACM Press, 2006.

[9] WATTS D J, STRONGATZ S H. Collective dynamics of ′ small - world′ networks[J]. Nature, 1998, 393(6): 440 - 442.