中国科学: 信息科学 2016 年 第 46 卷 第 2 期: 212–227
SCIENTIA SINICA Informationis
《中国科学》杂志社
SCIENCE CHINA PRESS
论 文

基于标签传播的可并行复杂网络重叠社区发现算法

李春英[①②], 汤庸[①*], 林海[①], 袁成哲[①], 麦辉强[①]

① 华南师范大学计算机学院, 广州 510631
② 广东技术师范学院计算机网络中心, 广州 510665
* 通信作者. E-mail: ytang@scnu.edu.cn

收稿日期: 2015–05–19; 接受日期: 2015–12–09; 网络出版日期: 2016–01–22
国家自然科学基金 (批准号: 61272067 和 61370229)、国家科技支撑计划 (批准号: 2013BAH72B01)、国家高技术研究发展计划 (863 计划) (批准号: 2013AA01A212)、广东省自然基金团队研究 (批准号: S2012030006242) 和广东省重大科技专项计划项目 (批准号: 2012A080104019) 资助项目

摘要 针对复杂网络重叠社区发现算法中预先输入参数的局限性以及标签冗余等缺点, 提出一种改进的复杂网络重叠社区发现算法, 最小极大团标签传播算法 MMCLPA (minimal maximal clique label propagation algorithm). 该算法通过寻找网络中的最小极大团 (MMC) 并对每个 MMC 中的节点赋予相同的标签来减少冗余标签, 提高算法的稳定性. 标签更新时采用亲密度作为权重并由 MMC 构成的核心节点群向四周扩散, 后期处理采用自适应阈值方式克服了预先输入参数对未知网络的局限性. 通过仿真和真实网络数据与其他几种有代表性的社区发现算法的实验对比分析, 表明 MMCLPA 算法提高了对混合参数 u 的容忍度以及算法的鲁棒性. 利用分布式计算模型 MapReduce 和 Hadoop 云平台实现了 MMCLPA 算法的并行化, 实验结果表明并行化的 MMCLPA 算法在百万级节点的复杂网络大数据中拥有单机系统下近似的社区发现质量以及良好的可扩展性.

关键词 复杂网络 社区发现 重叠社区 标签传播 并行计算

1 引言

复杂网络已成为计算机、物理、数学、生物、社会学及复杂性科学等多学科的研究热点 [1,2]. 复杂网络所拥有的一个最普遍和最重要的拓扑属性是社区结构, 即整个网络是由若干个群或社区构成. 一般我们认为社区是一组具有相似属性的节点集合, 并且在社区内部节点与节点之间连接紧密, 不同社区的节点间相互连接稀疏. 社区发现是利用网络的拓扑结构以及节点的信息, 通过特定有效的算法, 发现复杂网络中的社区结构以及每个节点所属的社区. 如在社交网络中, 社区发现的最终划分结果, 代表了真实的社会团体, 团体内部每个成员具有相同的兴趣、相似的行为. 在引文网络中, 通过社区发现, 每个团体都有相同的研究领域. 而在生物化学网络或者电子电路网络中, 最后划分出的社区是一些功能单元, 同一类功能单元在同一个社区 [3]. 发现这些网络中的社区, 有助于更好地理解和开发这些网

引用格式: 李春英, 汤庸, 林海, 等. 基于标签传播的可并行复杂网络重叠社区发现算法. 中国科学: 信息科学, 2016, 46: 212–227, doi: 10.1360/N112014-00258

络, 提供精确的个性化服务. 因此社区发现对网络结构分析、功能演化及预测有重要的理论意义和实际应用价值.

在许多现实世界的复杂网络中, 社区重叠是其中一个非常重要的特征 [4,5]. 近年来复杂网络重叠社区发现的相关研究也得到了越来越多的关注, 相关研究成果可以从多个角度进行阐述. 一是基于完全子图的算法, 2005 年 Palla 等 [6] 在 *Nature* 上发表了基于派系过滤的重叠社区发现算法 (CPM 算法), 该算法对传统社区模式进行扩展, 允许结点同时属于多个社区, 使重叠社区发现开始受到广泛关注, 并迅速成为社区发现的研究热点. 该算法认为社区结构由相邻的完全子图 (clique) 所构成, 一个节点可以属于多个 clique, 因而可以发现重叠社区结构. 经验表明该算法具有较高的时间复杂度, 在处理规模和密度较大的网络时往往需要较长时间或在有效时间内难以得到结果; 基于 clique 扩展的算法有 CPMw [7]、SCP [8] 以及 EAGLE [9] 等, 这类算法往往被认为更偏向于解决复杂网络中的模式匹配问题, 而非真正意义上的社区发现. 二是基于链接聚类的算法, 代表算法 LINK [10] 先对边集进行划分再转化为相应的社区结构, 针对 LINK 算法在复杂网络中准确度不高的特点, 比较有代表性的改进算法 Link Maximum Likelihood [11] 利用期望最大化来构建社区结构, Link-Comm [12] 利用信息论中的 Infomap 进行链接聚类, 虽然改进算法能够提高社区发现的准确度但可能出现过度重叠现象, 基于链接聚类的相关算法较多 [13~18]. 虽然基于链接聚类的方法是一种很自然的发现重叠社区的方法, 但是 Fortunato [19] 也指出基于边划分的方法并不能保证比基于节点的方法能够得到更好的社区发现质量. 三是基于局部优化和扩展的算法, 2009 年 Lancichinetti 等 [20] 提出了 LFM 算法, 它通过随机选取网络中的初始节点, 根据定义的局部适应度函数进行社区扩张, 直到适应度函数为负时停止, 由于各个社区的扩展过程相对独立, 因而每个节点可以被扩展到多个社区当中从而发现重叠社区结构. 但算法的适应度参数需要预先输入, 且初始种子节点对于局部社区的算法质量影响较大. 文献 [21 ∼ 28] 也是基于局部优化和扩展的相关算法. 四是基于标签传播的方法, Steve 将简单快速的标签传播算法 LPA (label propagation algorithm) [29] 进行了延伸, 提出了多标签传播算法 COPRA [30] 用以发现重叠社区, 该算法初始时每一个节点赋予一个标签并允许每个节点最多携带 v 个标签, 其继承了 LPA 算法时间复杂度低、适用大型网络等优点, 但对于未知的网络无法估计网络中节点所属的社区个数, 如果节点所属社区个数差别较大, 即使通过调节参数 v, 算法仍然难以发现较为准确的社区结构. 基于 LPA 算法简单高效的特点, 很多学者 [31~38] 从不同方面对其进行了改进, 但都存在结果不稳定等问题. 其他典型的重叠社区划分算法有基于中介度 (betweenness) 的分割算法 (GN 算法) [39], 该算法通过重复去掉具有最大中介度的边, 使得网络被分割成一些社区, 每去除一条边, 都需要重新计算中介度, 因此时间复杂度较大. 文献 [40] 综合封闭游走 (closed walks) 和聚类系数来取代 GN 算法中的边中介度概念, 通过迭代移除最低权值的边, 直到网络沦为孤立节点. 实验表明该算法较好地折衷了社区发现的精度和运行时间问题. 文献 [41] 提出了用模块度 (modularity) 作为社区划分质量的指标, 其原理是计算社区内的边数减去随机情况下的期望边数, 针对模块度有很多不同的改进优化及扩充方法 [42~48], 但对于社区大小差异较大的情形, 此类算法的结果却并不理想. 文献 [49] 提出基于顶点相似概率模型 (vertices similarity probability) 去发现未知复杂网络的社区结构. 基于同一社区的顶点具有相似的属性, VSP 模型使用顶点相似性发现社区结构, 采用矩阵摄动理论确定社区的数量, 经过仿真网络和真实网络的数据实验证明了该模型具有较好的性能, 但是像其他模块度方法一样, VSP 模型也存在着分辨极限问题. 文献 [50] 提出一种局部相似性度量, 对传统的 Ward 层次聚类算法进行推广, 用于发现复杂网络的社区结构. 该算法克服了传统局部相似性度量在某些情形下对节点相似性的低估倾向, 具有较好的可行性和有效性, 但其用于发现非重叠社区结构.

本文的贡献有两个方面: 一是针对基于标签传播的重叠社区发现算法的缺点, 提出了一种最小极

大团标签传播算法 MMCLPA (minimal maximal clique label propagation algorithm), 该算法提高了社区发现的质量以及算法的稳定性; 二是针对百万级节点的复杂网络大数据对单机系统带来的挑战, 利用分布式计算模型 MapReduce 和 Hadoop 云平台实现了 MMCLPA 算法的并行化, 使得对现实中普遍存在的超大规模复杂网络进行社区发现成为可能.

2 改进的标签传播算法 MMCLPA

定义 1 复杂网络由图 G=$\{V, E\}$表示, 其中 V 表示节点的集合, E 表示边的集合. 以 V 中未赋予标签的度数最大的节点 x 及其邻接节点中未赋予标签的度数最大的节点 y 为初始边寻找完全图 Gs, 则 Gs 称为团. 若 $Gs \subseteq G$, 且不存在任何团 $Gt \subseteq G$, 使得 $Gs \subset Gt$, 则称 Gs 为最小极大团, 简称 MMC.

定义 2 C_1 和 C_2 是两个不同的社区, 若节点 i 分别属于 2 个不同的社区, 则节点 i 为重叠节点, 即 $\text{overlap}(i) \Leftrightarrow \exists\, j\, \exists\, k\ (j,k \in V \wedge j \neq k \wedge (i,j) \in C_1 \wedge (i,k) \in C_2 \wedge C_1 \neq C_2)$.

定义 3 若叶子节点与邻居节点被划分到不同的社区, 则将叶子节点重新划分社区, 归属于邻居节点所属的最大社区中, 即 $\exists\, i \in V \wedge (i,j) \in E \wedge d(i) = 1 \wedge \text{overlap}(j) \in C_1 \cap C_2 \wedge C_1 \neq C_2 \wedge (|\ C_1\ | \neq |\ C_2\ |) \Rightarrow (i \in C_1 \wedge (|\ C_1\ | = \max(|\ C_1\ |, |\ C_2\ |))) \vee (i \in C_2 \wedge (|\ C_2\ | = \max(|\ C_1\ |, |\ C_2\ |)))$.

定义 4 存在一些社区, 是其他较大社区的子社区, 则将这些社区合并为一个社区, 即 $\exists\, C_i\, \exists\, C_j\ (C_i \neq C_j \wedge C_i \subseteq C_j) \Rightarrow C_i = (C_i \cup C_j) = C_j$.

基于标签传播的社区发现算法 MMCLPA 可以分成 3 个过程: (1) 标签初始化; (2) 标签更新; (3) 社区的形成及后期处理.

2.1 标签初始化

本文所指的社区是由最小极大团及其连通的节点组成的集合. 最小极大团是社区的核心单位, 处在同一个最小极大团中的节点, 必然处在同一个社区. 因此在标签初始化时, 关键在于寻找最小极大团. 算法为每一轮寻找到的最小极大团中的每一个节点赋予标签, 后续的寻找过程不再考虑这些节点, 可以大量减少冗余标签. 结合最小极大团的定义, MMCLPA 算法的标签初始化过程步骤如下所示, 算法伪代码如 Algorithm 1 所示.

(1) 对网络中的所有节点按度数进行降序排序, 令 t=1;

(2) 取未赋予过标签的度数最大的节点 A 及其邻居节点中未赋予过标签的度数最大的节点 B;

(3) 从 A 和 B 节点出发, 寻找网络中的最小极大团;

(4) 将标签 t 赋予这个最小极大团中的每一个节点;

(5) 设置 t=t+1;

(6) 重复步骤 (2)~(5), 直到没有满足要求的节点, 算法终止.

Algorithm 1 列出了 A 和 B 两个节点寻找最小极大团并为团内每一个节点赋予标签过程, 节点 A, B 为未赋予标签节点中两个度数最大的相邻节点. 在 FindMinMaxCliquesForTwoNodes 算法中方法 isLastElement 判断是否是交集的最后一个元素, isInClique 判断节点是否和 cliques 组成团. 若和 cliques 的每个节点都存在边, 则将此节点也加入到 cliques 中, isLabeled 判断该节点是否已经赋予过标签, setLabelForCliques 表示为通过两个节点找到的最小极大团赋予标签. 初始化过程后一些节点拥

Algorithm 1: 寻找 MMC 并为其内节点赋予标签

Input:网络中未赋予标签的两个度数最大的相邻节点

Output:最小极大团及其对应的标签

```
1  Procedure FindMinMaxCliquesForTwoNodes (nodeA, nodeB);
2  intersection=nodeA.neighbors ∩ nodeB.neighbors;
3  for each node in intersection do
4    if !node.isLabeled() then
5      Stackstack = new Stack();
6      stack.push (node);
7      while !stack.isEmpty() do
8        if node.isLastElement() then
9          SetcliqueSet = getNodeSetFromStack (stack);
10           booleanisInClique = isInClique (nextnode, cliqueSet);
11           if isInClique() then
12             cliqueSet.add (node);
13             stack.pop();
14             else
15               Set cliqueSet = getNodeSetFromStack (stack);
16               boolean isInClique = isInClique (node, cliqueSet);
17               if isInClique() then
18                 stack.push (node);
19       end while
20 end for
21 setLabelForCliques();
22 end Procedure
```

有了标签队列, 非 MMC 中的节点则没有获得标签. 根据复杂网络小世界模型, 只要节点拥有邻居, 则在标签更新规则的算法迭代中可使其拥有标签.

根据 MMCLPA 标签初始化规则, 图 1 为原始网络拓扑结构图, 图 2 为经过标签初始化后得到的网络拓扑结构图, 根据图 1 的拓扑结构利用 Algorithm 1 在图 2 中共找出 3 个最小极大团 MMC, 分别为节点群 (4,6,7,8), (1,2,3,5,14) 和 (10,11,12,15), 并分别为每一个 MMC 中的节点赋予相同的标签.

2.2 标签更新规则

通过对大量复杂网络拓扑结构分析得知, 每一个社区至少有一个影响力比较大的核心节点群, 社区的拓扑关系则由核心节点群向社区边缘扩散, 而此核心节点群即为本文定义的最小极大团 MMC. 标签在传播时, 采用从核心节点群 MMC 向外围扩展, 则能提高算法的稳定性和效率. 另外, 节点是否属于某个社区则与节点的邻接节点以及亲密度有关, 节点是否为重叠节点则与节点在多个标签中的度数分布有关. 具体操作步骤如下所述.

(1) 将标签初始化后的最小极大团中的每一个节点的标签权重设置为 1;

(2) 统计其他节点与 MMC 节点相邻的亲密度 $F/d(v_i)$, 其中 F 为节点 v_i 与 MMC 相邻的次数, $d(v_i)$ 为节点 v_i 的度数, 该亲密度即为节点 v_i 在 MMC 标签下的权重;

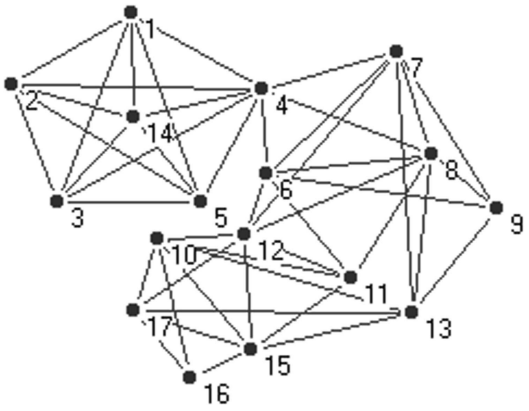

图 1 复杂网络原始图

Figure 1 Complex network original figure

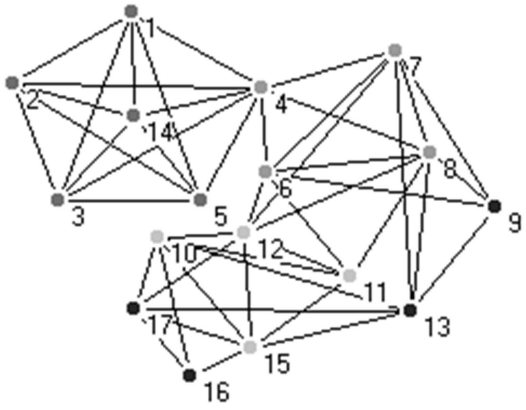

图 2 复杂网络初始化结果图

Figure 2 Complex network initialized figure

表 1 标签更新过程

Table 1 Process of the label propagation

MMC ID/Label	MMC node	MMC N-node	Intimacy	Label	Weight
1	4,6,7,8	1	1/5	1	1/5
		2	1/5	1	1/5
		3	1/5	1	1/5
		5	1/5	1	1/5
		14	1/5	1	1/5
		12	3/7	1	3/7
		11	2/5	1	2/5
		13	2/6	1	2/6
		9	3/4	1	3/4
2	1,2,3,5,14	4	5/8	2	5/8
3	10,11,12,15	6	2/6	3	2/6
		7	1/6	3	1/6
		8	2/7	3	2/7
		13	2/6	3	2/6
		17	3/5	3	3/5
		16	2/3	3	2/3

(3) 将非 MMC 及其相邻的节点随机排列, 对每一个节点进行标签更新, 节点拥有其所有邻居节点的标签号, 每个标签号的权重为其在邻居节点中权重最大的一个.

对图 1 所示的复杂网络标签初始化后的结果按照标签更新规则更新节点的标签及权重, 结果如表 1 所示. 表中 MMC node 表示 MMC 中的节点, MMC N-node 表示与 MMC 相邻的节点, Intimacy 为 MMC N-node 节点与 MMC 的亲密度. 标签更新规则算法的伪代码如 Algorithm 2 所示.

在 Algorithm 2 中, nodeList 是全量节点集合, cliqueCollection 是全量最小极大团集合, pendingNodeList 为待处理节点列表, 函数 subCliqueByNodeList.Size 判断当前节点是否存在于 MMC 中,

Algorithm 2: 标签更新过程伪代码

```
Input:网络节点, 部分标签及对应的权重
Output:网络节点, 标签及对应的权重
Procedure UpdateLableAndWeightForNode (nodeList,cliqueCollection);
List pendingNodeList = nodeList.clone();
处理步骤 1,2;
for each node in nodeList do
  List subCliqueByNodeList = cliqueCollection.getCliquelByNode (node);
  if subCliqueByNodeList.size()>0 then
    for each clique in subCliqueByNodeList do
      Double weight = 1;
      saveResult (node,clique.lable,weight);
    pendingNodeList.removeIfExist (node);
    end for
    List allCliqueList=cliqueCollection.getAllCliquel();
    for each clique in allCliqueList do
      Int intimacy = clique.getIntimacyForNode (node);
      If intimacy>0 then
        Double weight = intimacy/node.getAllneighborCount();
        saveResult (node,clique.lable,weight);
        pendingNodeList. removeIfExist (node);
    end for
end for
处理步骤 3;
while pendingNodeList.isNotEmpty() do
  for each node in pendingNodeList do
    List neighborList = node.getNeighbors();
    for each neighborNode in neighborList do
      if checkInResult (neighborNode) then
        Double weight = neighborNode.getMaxWeight();
        saveResult (node,neighborNode.lable,weight);
        pendingNodeList.removeIfExist (node);
    end for
  end for
  pendingNodeList.converse();
end while
end Procedure
```

saveResult 函数输出 node 节点、标签及其权重, pendingNodeList.converse 的作用是每处理一次后 list 内容颠倒过来, 防止 while 死循环.

表 2　标签更新后期处理过程

Table 2　Label's postprocessing

Node	Label (Weight)	W-threshold $(1/(L+1))$	Result	Community
1	2(1), 1(1/5)	1/3	2(1)	C_2
2	2(1), 1(1/5)	1/3	2(1)	C_2
3	2(1), 1(1/5)	1/3	2(1)	C_2
4	1(1), 2(5/8)	1/3	1(1), 2(5/8)	C_1C_2
5	2(1), 1(1/5)	1/3	2(1)	C_2
6	1(1), 3(2/7)	1/3	1(1)	C_1
7	1(1), 3(1/6)	1/3	1(1)	C_1
8	1(1), 3(2/7)	1/3	1(1)	C_1
9	1(3/4)	1/2	1(3/4)	C_1
10	3(1)	1/2	3(1)	C_3
11	3(1), 1(2/5)	1/3	3(1), 1(2/5)	C_1C_3
12	3(1), 1(3/7)	1/3	3(1), 1(3/7)	C_1C_3
13	1(2/6), 3(2/6)	1/3	1(2/6), 3(2/6)	C_1C_3
14	2(1), 1(1/5)	1/3	2(1)	C_2
15	3(1)	1/2	3(1)	C_3
16	3(2/3)	1/2	3(2/3)	C_3
17	3(3/5)	1/2	3(3/5)	C_3

2.3　后期处理规则

针对 COPRA 算法中预先输入重叠社区个数在未知网络社区发现中存在的问题, 本文经过深入分析网络节点标签权重与社区个数的关系, 采用自适应阈值对标签传播后的节点标签队列进行后期处理, 每一个标签代表一个社区, 标签相同的节点组成一个社区. 具体过程如下所示.

(1) 当节点 v_i 拥有多个标签时, 统计该节点的标签个数 L, 删除权重小于阈值 $1/(L+1)$ 的标签; 若所有标签权重都小于 $1/(L+1)$, 则保留权重最大的标签, 删除其他所有标签, 如果有多个权重最大, 同时又都小于 $1/(L+1)$ 的标签时, 随机选择一个标签保留, 删除其他所有的标签;

(2) 若存在一些社区, 是其他较大社区的子社区, 则将该子社区与较大的社区合并;

(3) 只有一个标签的节点归为该标签所属的唯一社区中, 拥有多个标签的节点则属于多个社区.

对表 1 的数据按照标签更新后期处理规则进行标签后期处理, 过程如表 2 所示. 其中 W-threshold $(1/(L+1))$ 表示节点的自适应阈值, Result 表示通过自适应阈值处理后的结果. 从结果可以看出社区 C_1 中节点为 (4,8,7,6,9,11,12,13), 社区 C_2 中节点为 (1,2,3,4,5,14), 社区 C_3 中节点为 (12,15,10,11,13,16,17), 故具有 2 个社区的重叠节点分别为 (4,11,12,13), 与实际情况相符.

2.4　MMCLPA 的时间复杂度

假设网络中有 n 个节点且平均度数为 k, 最小极大团数目为 h, 最小极大团中节点平均个数为 m. 则在标签初始化阶段, 算法的时间复杂度近似为 $O((kn/2)(n/2+5))$, 在标签传播阶段已经拥有标签的节点个数近似为 hm, 与最小极大团相邻的节点个数近似为 khm, 通过第一轮标签传播使 khm 个节点获得标签的时间复杂度为 $O(khm^2)$. 剩余 $(n-hm-khm)$ 个节点通过两层循环获得标签, 外层是标

签传播的迭代次数, 一般值为 5∼7, 内层循环是网络中未获得标签的 $(n-hm-khm)$ 个节点, 内层循环中每个节点接受一个标签分两个过程, 一是接受每个邻居节点的新标签, 另一个是对与邻居节点相同的标签进行比较, 选择权重最大的标签. 第一个过程是在邻居节点标签队列中选择标签, 时间复杂度为 $O(1)$; 第二个过程取决于节点的度数, 时间复杂度为 $O(k)$. 因此, 标签传播过程的时间复杂度为 $O(T(n-hm-khm)k)$, MMCLPA 算法的时间复杂度为 $O(k(n^2/4+5n/2+hm^2+T(n-hm-khm)))$. 对于复杂网络大数据而言, 该算法时间复杂度较大, 后文采用将 MMCLPA 算法并行化的方案解决了算法复杂度较高的问题.

3 算法实验

为验证 MMCLPA 算法的性能, 我们将 MMCLPA 与其他几种具有代表性的重叠社区发现算法进行比较, 待比较的算法分别是基于模块度优化的算法 LFM, 基于派系过滤算法 CPM 的实现程序 CFinder, 基于标签传播的重叠社区发现算法 COPRA, 对标签传播算法 LPA 改进的算法 SLPA.

3.1 实验数据与评价

实验采用了两种数据集, 一种是仿真网络数据集, 另一种是真实网络数据集. 采用 LFR 基准程序 [51] 来生成仿真网络数据集, 该程序可以生成具有真实网络特性的仿真网络, 并且很多参数是可以调节的, 如节点数、平均度数、社区大小等. 另外, 在生成仿真网络的同时, 程序也会把相应的社区结构输出, 因此很适合用来验证社区发现算法. 本文采用 NMI (标准互信息) 指标来衡量仿真网络社区划分结果的质量. NMI 指标的公式如式 (1) 所示. 其中, A 和 B 表示网络的两种划分结果, $\boldsymbol{C}$ 表示混合矩阵, 其元素 C_{ij} 表示划分 A 中的社团 i 里面的节点在划分 B 中的社团 j 里面出现的个数. C_A 和 C_B 表示划分 A 和划分 B 中的社团的个数, C_i, 表示矩阵 $\boldsymbol{C}$ 中第 i 行的元素之和, $C_{,j}$ 表示矩阵 $\boldsymbol{C}$ 中第 j 列的元素之和, n 表示网络的节点个数. NMI 值为 $0\sim1$ 之间的数, NMI 值越大表示两种划分越相似.

$$\mathrm{NMI}=\frac{-2\sum_{i=1}^{C_A}\sum_{j=1}^{C_B}C_{ij}\log\left(\dfrac{{C_{ij}}^n}{C_i,C_{,j}}\right)}{\sum_{i=1}^{C_A}C_{i,}\log\left(\dfrac{C_i,}{n}\right)+\sum_{j=1}^{C_B}C_{,j}\log\left(\dfrac{C_{,j}}{n}\right)}. \tag{1}$$

大多数社区发现算法都以真实网络来验证算法的质量和效率, 对于真实网络, 如 karate, football, dolphin, Enren, CA-HepPH, BJTU 论文合作网络等, 这些网络的规模大小不一, 利用这些真实网络数据, 采用评价重叠社区函数 EQ 衡量社区的发现质量, 实现与其他算法性能的比较. EQ 函数如式 (2) 所示. 其中, $\boldsymbol{A}$ 是网络的邻接矩阵, O_v 表示节点 v 属于的社区的数量, m 表示网络中边的个数, k_v 表示节点 v 的度, 另外, 如果所有的节点都在一个社区中, EQ 函数的值为 0.

$$EQ=\frac{1}{2m}\sum_i\sum_{v,w\in C_i}\frac{1}{o_v o_w}\left[\boldsymbol{A}_{vw}-\frac{k_v k_w}{2m}\right]. \tag{2}$$

3.2 仿真网络实验

为了客观的评价不同社区发现算法的划分质量, 从规模和边的稠密度出发设计了 4 种类型的仿真网络, 分别是 S_1 表示稀疏网络小社区, S_2 表示稀疏网络大社区, S_3 表示稠密网络小社区, S_4 表示稠

表 3 LFR 基准网络数据

Table 3 LFR benchmark network data

Network ID	n	k	maxk	minc	maxc	t_1	t_2	O_n	O_m	u
S_1	5000	20	100	20	100	2	2	100	4	0.1∼0.8
S_2	5000	20	100	50	500	2	2	100	4	0.1∼0.8
S_3	5000	40	100	20	100	2	2	100	4	0.1∼0.8
S_4	5000	40	100	50	500	2	2	100	4	0.1∼0.8

密网络大社区, 如表 3 所示. 其中 n 是网络的节点数, k 是节点的平均度数, maxk 是节点的最大度数, maxc 是一个社区的最大节点数, minc 是一个社区的最小节点数, t_1 是节点度数幂率分布指数, t_2 是社区大小幂率分布指数, O_n 是重叠节点的个数, O_m 是重叠节点属于的社区个数, u 是混合参数, 其取值范围是 $[0,1]$, 取值越大生成的仿真网络社区结构越不明显, 当取值为 0 时整个网络是一个社区, 取值为 1 时则是一个随机网络. 实验通过调整混合参数 u 来控制网络的社区结构, $u \in [0.1, 0.8]$, 每次增加 0.1, 从而计算在网络不同结构下各算法的 NMI 值.

各个算法的参数设置如下: COPRA 中的 v 值分别设置为 $2 \sim 5$; Cfinder 中的 k 分别设置为 $3 \sim 8$; LFM 中的 α 分别设置为 $0.8 \sim 1.6$. 每种算法从不同参数下的 NMI 值中选取最大值作为最终的结果. 图 3 为 5 种典型算法分别在 4 组 LFR 基准网络数据上的社区发现结果, 将 MMCLPA 算法分别与其他几种算法进行结果分析比较.

(1) 从实验结果可知, SLPA 和 COPRA 算法的社区发现结果呈现了震荡的趋势. 这与我们的分析一致, COPRA 和 SLPA 算法因标签初始化时为每一个节点赋予唯一标签导致自身拥有过多冗余标签, 以致于算法不稳定; 与 SLPA 和 COPRA 算法相比, MMCLPA 算法减少冗余标签率可达 75%, 因此在标签传播时减少了很多不必要的判断开销, 提高了算法效率, 算法结果也更稳定.

(2) 在混合参数 $u \in [0.1, 0.5]$ 时, 大多数算法具有较好的社区划分质量, 随着 u 的增大, 社区结构变得不明显, 要准确分离网络中预设的社团将变得更加困难; 当 u>0.5 时, 大多数算法的划分质量出现了明显的下降, 但 MMCLPA 的划分质量仍保持在较高水平, 在稀疏 (稠密) 网络社区中取得了较好的划分质量, 因此该算法可以提高标签传播算法对混合参数 u 的容忍度, 发现一些社区结构较不明显的社区.

(3) 在稠密的网络中 CFinder 和 LFM 算法不具备较好的重叠社区识别能力, 不能处理规模较大的复杂网络.

(4) 与其他四种算法相比, MMCLPA 算法在混合参数 u 值逐渐增加, 社区发现难度逐渐增大的情况下, 在稀疏 (稠密) 网络中均呈现了较好的社区发现效果, 并且在稀疏网络中的结果好于稠密网络, 这与我们对网络拓扑结构的深度分析以及最小极大团的定义有关. 在稀疏网络中或者当网络的社区结构不明显时, 标签初始化算法仍可以发现至少拥有 3 个节点的拓扑结构 (MMC), 使其作为核心节点群并根据标签更新规则进行节点的标签更新进而达到社区发现的目的.

综上, 4 种类型仿真网络的实验结果表明 MMCLPA 算法大量减少了冗余标签, 提高了对参数 u 的容忍度, 可以对社区结构较不明显的复杂网络进行社区发现, 具有较好的社区发现效果.

3.3 真实网络实验

Karateclub 是圣所迦利的空手道俱乐部的人际关系图, 具有 34 个节点和 78 条边的小规模真实网络. CA-HepPH 是一个科学家合作的较大规模真实网络, 具有 12008 个节点和 237010 条边. 这两个

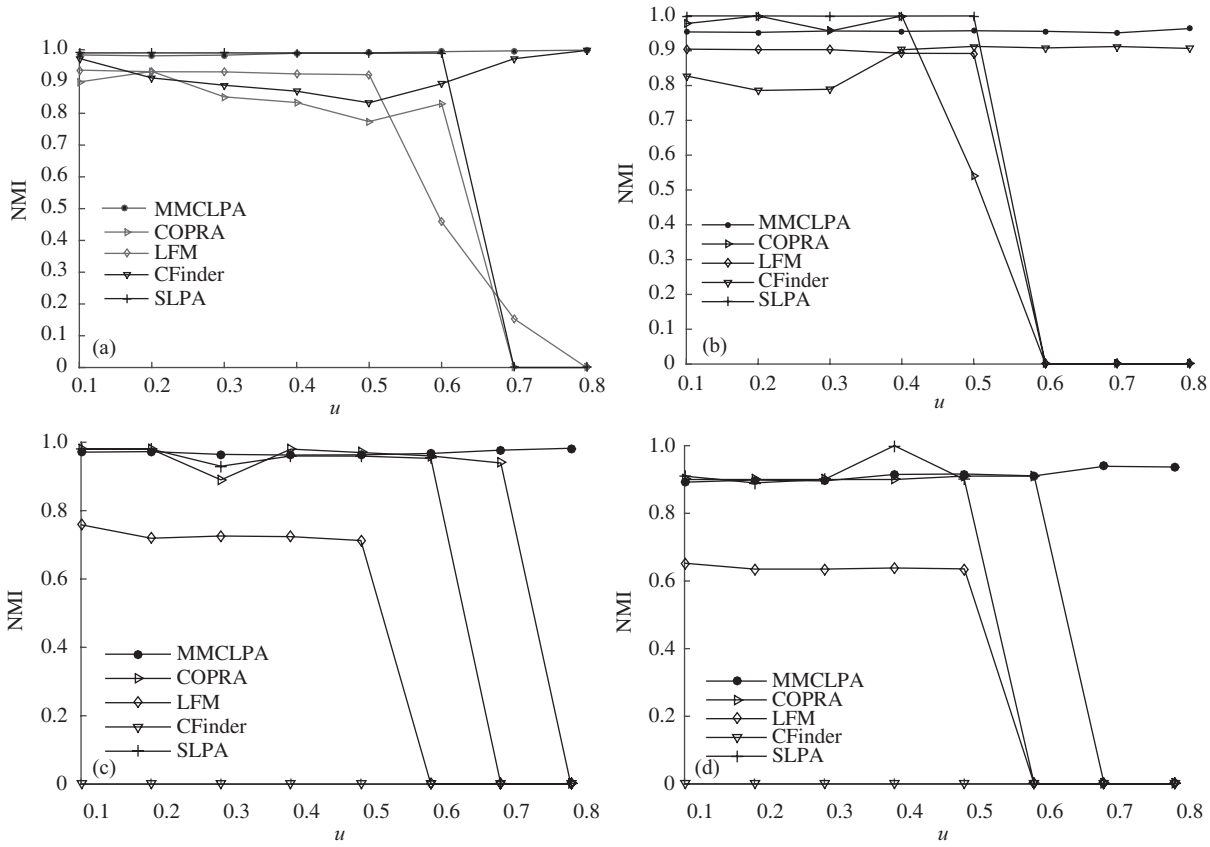

图 3 4 种类型仿真网络上的社区发现结果

Figure 3 Result of community detection on four kinds of benchmark network data. MMCLPA on S_1 (a), S_2 (b), S_3 (c) and S_4 (d)

真实网络中的节点代表人, 而边代表了人与人之间的关系. 在真实网络上的实验, 各种算法分别选取在不同参数下的最大模块度 EQ 值作为最终的结果. 表 4 显示了 5 种算法分别在 2 个真实网络上的 EQ 最大值以及对应的结果方差 (R-variance). 从实验结果可以得知, 算法 COPRA 和 SLPA 结果方差最大, 说明算法的稳定性较差, 而 MMCLPA 算法在两种真实网络数据集上均取得了较高的 EQ 值, 同时保证了较小的结果方差. 因此验证了 MMCLPA 算法具有较好的稳定性.

4 MMCLPA 算法的 MapReduce 实现

4.1 算法实现

MapReduce 编程模型虽然简单, 但其数据应满足局部相关性. MMCLPA 算法在求最小极大团以及标签传播过程中, 每个节点只与其邻居节点有关, 满足数据的局部相关性. 因此, 最初整个无向无权重网络数据可以分散到不同的机器节点中, 使用一轮 MapReduce 过程, 将整个网络数据转换为邻接表文件, 使每个节点都包含其邻居节点信息, 利用 MapReduce 迭代来实现标签传播, 每一轮 MapReduce 是一次标签传播过程. 因此可以利用 MapReduce 实现 MMCLPA 算法的并行化.

MMCLPA 算法中一个节点可以拥有多个标签, 因此并行化的关键在于利用 MapReduce 找到网

表 4　两个真实网络上的社区发现结果

Table 4　Result of community detection on the two real network data

E-methods	Karateclub real network					CA-HepPH real network				
	COPRA	MMCLPA	SLPA	LFM	CFinder	COPRA	MMCLPA	SLPA	LFM	CFinder
EQ	0.72	0.81	0.76	0.65	0.73	0.7	0.78	0.75	0.58	0.61
R-variance	0.18	0.02	0.15	0.1	0.13	0.25	0.07	0.21	0.13	0.15

Algorithm 3: MMCLPA 并行化实现的伪代码

Input:key 为开始节点, value 为所有与开始节点存在边的邻居节点

Output:输出节点新标签到 redis 中

```
1  InitLabelMapper();
2  InitLabelReducer();
3  Procedure LabelPropagationMapper (key, value)
4  (nodeID, neighborsID) = parse (value);
5  nodeLabel = redis.get (nodeID);
6  list labels = empty;
7  for each neighborID in neighborsID do
8     labels.add (redis.get (neighborID));
9  label = maxLabelInLabels (labels);
10  redis.push < nodeID, label >;
11  end Procedure;
```

络中的最小极大团以及存取节点的标签信息. 算法实现首先解析所有与开始节点存在边的邻居节点得到节点的邻居信息, 按度数降序排序后以队列形式存储在 redis 中, 将网络的边文件转换为邻接表文件并存储在 HDFS 中, 采用 redis 完成邻居节点的交集, 寻找 MMC 并为每个 MMC 中的节点赋予同一个标签. 通过 InitLabelMapper 和 InitLabelReducer 两个过程完成了标签的初始化, 采用 redis 记录每个节点及其对应标签以及按照规则对标签进行实时异步更新. 算法的 MapReduce 实现主要步骤如下所示，对应的伪代码如 Algorithm 3 所示.

(1) 将边文件存储在 HDFS 中, 经过一轮 MapReduce 将边文件转换为网络的邻接表文件 NodeNeighborsFile, 并将节点的邻接表存储在 Redis 中;

(2) 以 NodeNeighborsFile 为输入, 借助 redis, 经过一轮 MapReduce 找到节点的最小极大团, 完成标签的初始化, 并将节点的标签信息存入到 redis 中;

(3) 借助 redis 以 NodeNeighborsFile 作为输入, 经过多轮 MapReduce 完成标签的传播, 每一轮 MapReduce 是标签的一次传播.

在 Algorithm 3 中利用函数 maxLabelInLabels() 保存邻居节点每个标签的权重最大值以及相应标签, 所有的节点 — 标签对均存储在 redis 当中. 一次 LabelPropagationMapper 任务对应一轮 MMCLPA 算法迭代过程, 多次运行 LabelPropagationMapper 任务使所有节点均获得稳定的标签队列, 经过后期处理得到社区发现的结果.

表 5 MMCLPA 并行化算法的 LFM 网络数据

Table 5 Benchmark network data of parallelize MMCLPA

Network ID	n(million)	k	maxk	minc	maxc	t_1	t_2	O_n	O_m	u
P1	1∼10	40	100	50∼500	200∼2000	2	2	200∼2000	4	0.2
P2	0.01∼0.1	20	100	20∼100	200∼500	2	2	50∼100	4	0.2

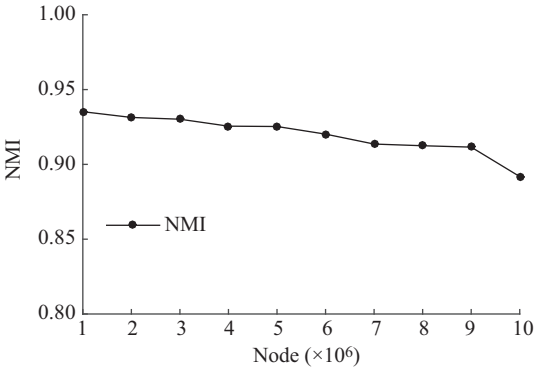

图 4 并行化 MMCLPA 算法在不同数据规模上的社区发现结果

Figure 4 Community detection of parallel MMCLPA on the different scale of Benchmark network data

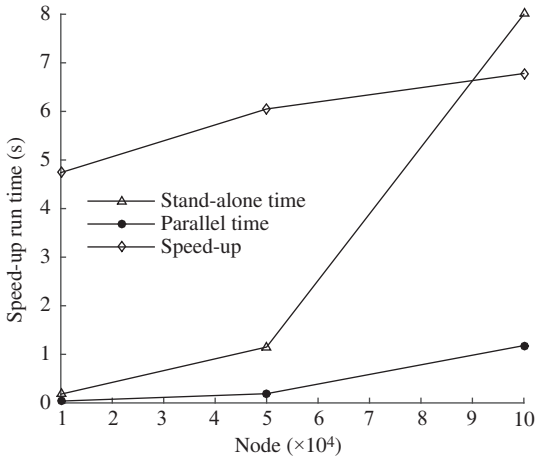

图 5 并行化 MMCLPA 算法在不同网络数据规模上的加速比

Figure 5 Speed-up of parallel MMCLPA on the different scale of Benchmark network data

4.2 实验结果

实验所用的 Hadoop 集群包括 1 个主节点 7 个从节点. 硬件环境为 4 核 3.2 GHz 处理器, 16 GB 内存. 操作系统为 DebianGNU/Linux64 位服务器, 内核版本为 2.6.32. 实验依然采用 LFR 基准程序来生成仿真网络数据. 具体参数如表 5 所示.

在 P1 网络数据集下实验得到的 NMI 值如图 4 所示, 在混合参数 u=0.2 不变的情况下从实验结果可以看出, 随着数据规模的增长, MMCLPA 并行算法的社区发现质量也出现了一定程度的下降, 但总体保持超过 80% 的正确率, 与在单机系统下 u=0.2 时处理小规模数据集的社区发现质量近似, 说明 MMCLPA 并行化算法与单机系统下的 MMCLPA 算法具有近似的算法质量. 为了更好的验证 MMCLPA 算法的并行化性能, 且由于单机系统的性能有限以及能在有效时间内完成对比实验, 实验中使用了相对较小的 P2 网络数据集, 将 MMCLPA 算法在单机系统上的运行时间与上文我们构建的 Hadoop 集群系统上的运行时间进行比较, 结果如图 5 所示. 图 5 展示了 MMCLPA 算法在不同数据规模下并行和单机系统中运行所需的时间以及加速比性能, 其中为了能够较好地展示运行时间和加速比, 图 5 中展示的运行时间是各个结果值的 1/100. 从图中可以看出 MMCLPA 可以处理大规模的数据集. 并且随着数据规模的增长 MMCLPA 算法在单机系统下的运行时间近似指数的增长, 而并行化的处理时间则明显减少. 从图中的加速比曲线可以看出当数据规模较小时, 并行处理并未得到很好的加速比性能, 这是因为 Map Reduce 的每个操作都会有 I/O 时间损耗, 当数据规模较小时, 损耗所占的比重增大. 随着数据规模的增大, I/O 时间损耗比重减小, 并行化性能获得提升, 因此 MMCLPA 并

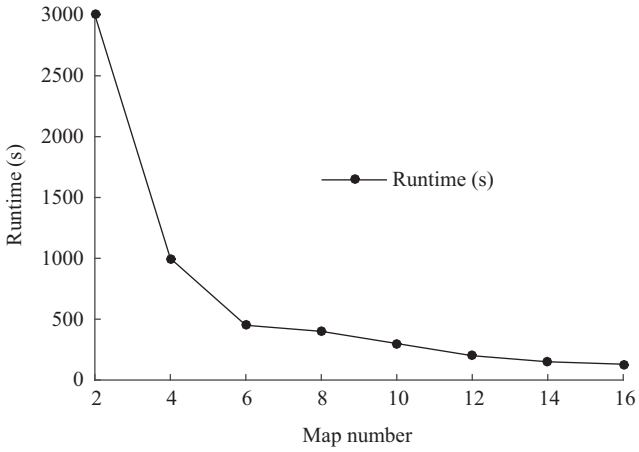

图 6　并行化 MMCLPA 算法在不同 map 数目上的运行时间图

Figure 6　Runtime of parallel MMCLPA on the different amounts of maps

行化算法提高了对数据的容纳度. 另外在固定数据集大小为 1 百万节点时进行实验, 得出不同的 Map 数目与算法的运行时间如图 6 所示. 可以看出增加 Map 和 Reduce 的数目可以减少算法的运行时间. 也就是说参与运算的机器越多耗时越少. 因此 MMCLPA 并行化算法具有良好的可扩展性.

5　结束语

本文提出了一种改进的基于标签传播的社区发现算法, 通过考虑网络中的拓扑结构, 获取网络中的最小极大团, 将最小极大团作为社区的核心节点群, 利用其他节点与最小极大团 MMC 的亲密度作为标签的权重并采用邻接节点标签权重最大值的方式进行标签传播, 采用自适应阈值的后期处理方式摒弃了预先输入社区个数对未知网络社区划分质量的影响. 相比于每个节点分配唯一一个标签的重叠社区发现算法, MMCLPA 算法可以在标签传播时减少不必要的判断开销, 提高算法的效率, 并在一定程度上提高了算法的稳定性. 通过仿真网络和真实网络的数据实验表明, 在社区结构较为明显的网络环境 ($u \leqslant 0.5$) 中具有较好的社区发现质量, 同时该算法在社区结构相对不明显的复杂网络 ($u > 0.5$) 中社区发现质量仍好于其他几种典型的重叠社区发现算法. 因此, MMCLPA 算法提高了对混合参数 u 的容忍度以及算法的稳定性. 另外, 利用 MapReduce 编程模型和 Hadoop 云平台, 将 MapReduce 分布计算模型与 MMCLPA 算法相结合, 解决了 MMCLPA 的并行化实现问题. 通过在百万级节点的复杂网络大数据上的实验表明基于 MapReduce 编程模型的 MMCLPA 算法, 拥有单机系统下近似的社区发现质量以及良好的可扩展性. 因此并行化的 MMCLPA 算法解决了该算法复杂度较高的问题, 也满足了复杂网络大数据的社区发现需求.

参考文献

1　He D X, Zhou X, Wang Z, et al. Community mining in complex networks clustering combination based genetic algorithm. Acta Automatica Sinica, 2010, 36: 1160–1170 [何东晓, 周栩, 王佐, 等. 复杂网络社区挖掘 —— 基于聚类融合的遗传算法. 自动化学报, 2010, 36: 1160–1170]

2　Scheffer M. Complex systems: foreseeing tipping points. Nature, 2010, 467: 411–412

3　Fortunato S. Community detection in graphs. Phys Rep, 2010, 486: 75–174

4　My T T, Panos M P. Handbook of Optimization in Complex Networks. Berlin: Springer, 2011

5 Reid F, McDaid A F, Hurley N J. Partitioning breaks communities. In: Proceedings of International Conference on Advances in Social Networks Analysis and Mining, Kaohsiung, 2011. 79–105

6 Palla G, Derenyi I, Farkas I, et al. Uncovering the overlapping community structure of complex networks in nature and society. Nature. 2005, 435: 814–818

7 Farkas I J, Ábel D, Palla G, et al. Weighted network modules. New J Phys, 2007, 9: 180–198

8 Kumpula J M, Kivel M, Kaski K, et al. Sequential algorithm for fast clique percolation. Phys Rev E, 2008, 78: 026109

9 Shen H W, Cheng X Q, Cai K, et al. Detect overlapping and hierarchical community structure in networks. Phys A: Stat Mech Appl, 2009, 388: 1706–1712

10 Ahn Y Y, Bagrow J P, Lehmann S. Link communities reveal multiscale complexity in networks. Nature, 2010, 466: 761–764

11 Ball B, Karrer B, Newman M E J. An efficient and principled method for detecting communities in networks. Phys Rev E, 2011, 84: 036103

12 Kim Y, Jeong H. Map equation for link community. Phys Rev E, 2011, 84: 026110

13 Shen H W, Cheng X Q, Guo J F. Quantifying and identifying the overlapping community structure in networks. J Stat Mech: Theory Exp, 2009, 2009: P07042

14 Fu X H, Liu L D, Wang C. Detection of community overlap according to belief propagation and conflict. Phys A: Stat Mech Appl, 2013, 392: 941–952

15 Zhang Y, Yeung D Y. Overlapping community detection via bounded nonnegative matrix tri–factorization. In: Proceedings of the 18th ACM SIGKDD Int Conf on Knowledge Discovery and Data Mining, Beijing, 2012. 606–614

16 Lin Y F, Wang T Y, Tang R, et al. An effective model and algorithm for community detection in social networks. J Comput Res Dev, 2012, 49: 337–345 [林友芳, 王天宇, 唐锐, 等. 一种有效的社会网络社区发现模型和算法. 计算机研究与发展, 2012, 49: 337–345]

17 Zhu M, Meng F R, Zhou Y. Density-based link clustering algorithm for overlapping community detection. J Comput Res Dev, 2013, 50: 2520–2530 [朱牧, 孟凡荣, 周勇. 基于链接密度聚类的重叠社区发现算法. 计算机研究与发展, 2013, 50: 2520–2530]

18 Evans T S, Lambiotte R. Line graphs, link partitions and overlapping communities. Phys Rev E, 2009, 80: 016105

19 Fortunato S. Community detection in graphs. Phys Rep, 2010, 486: 75–174

20 Lancichinetti A, Fortunato S, Kertész J. Detecting the overlapping and hierarchical community structure in complex networks. New J Phys. 2009, 11: 033015

21 Kelley S. The existence and discovery of overlapping communities in large–scale networks. Dissertation for Ph.D. Degree. Troy: Rensselaer Polytechnic Institute, 2009

22 Havemann F, Heinz M, Struck A, et al. Identification of overlapping communities and their hierarchy by locally calculating community–changing resolution levels. J Stat Mech: Theory Exp, 2011, 2011: 01023

23 Jin D, Yang B, Baquero C, et al. A markov random walk under constraint for discovering overlapping communities in complex networks. J Stat Mech: Theory Exp, 2011, 2011: 05031

24 Arnau P S, Guillem P L, Julián P, et al. Overlapping community search for social networks. In: Proceedings of the 26th International Conference on Data Engineering (ICDE), Long Beach, 2010. 992–995

25 Chen D B, Shang M S, Lv Z H, et al. Detecting overlapping communities of weighted networks via a local algorithm. Phys A, 2010, 389: 4177–4187

26 Cazabet R, Amblard F, Hanachi C. Detection of overlapping communities in dynamical social networks. In: Proceedings of the 2ed conference on social computing, Minneapolis, 2010. 309–314

27 Lancichinetti A, Radicchi F, Ramasco J J, et al. Finding statistically significant communities in networks. PLoS One, 2011, 6: e18961

28 Coscia M, Rossetti G, Giannotti F, et al. DEMON: a local-first discovery method for overlapping communities. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, 2012. 615–623

29 Raghavan U, Albert R, Kumara S. Near linear time algorithm to detect community structures in large-scale networks. Phys Rev E, 2007, 76: 036106

30 Gregory S. Finding overlapping communities in networks by label propagation. New J Phys, 2010, 12: 103018

31 Xie J R, Szymanski B K. Towards linear time overlapping community detection in social networks. In: Proceedings of

the 16th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Kuala Lumpur, 2012. 25–36
32 Liu X, Murata T. Advanced modularity-specialized label propagation algorithm for detecting communities in networks. Phys A: Stat Mech Appl, 2010, 389: 1493–1500
33 Jin D, Liu J, Yang B, et al. Genetic algorithm with local search for community detection in large-scale complex networks. Acta Automatica Sinica, 2011, 37: 873–882 [金弟, 刘杰, 杨博, 等. 局部搜索与遗传算法结合的大规模复杂网络社区探测. 自动化学报, 2011, 37: 873–882]
34 Xie J R, Szymanski B K. Community detection using a neighborhood strength driven label propagation algorithm. In: Proceedings of IEEE Network Science Workshop (NSW), New York, 2011. 188–195
35 Cordasco G, Gargano L. Community detection via semi–synchronous label propagation algorithms. In: Proceedings of 2010 IEEE International Workshop on Business Applications of Social Network Analysis (BASNA), Bangalore, 2010. 1–8
36 Raghavan U N, Albert R, Kumara S. Near linear time algorithm to detect community structures in large-scale networks. Phys Rev E, 2007, 76: 036106
37 Lancichinetti A, Fortunato S, Radicchi F. Benchmark graphs for testing community detection algorithms. Phys Rev E, 2008, 78: 046110
38 Lancichinetti A, Fortunato S. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. Phys Rev E, 2009, 80: 016118
39 Girvan M, Newman M E J. Community structure in social and biological networks. Proceedings of the National Academy of Sciences, 2002, 99: 7821–7826
40 Yang Y, Sun P G, Xia H, et al. Closed walks for community detection. Phys A: Stat Mech Appl, 2014, 397: 129–143
41 Newman M E J, Girvan M. Finding and evaluating community structure in networks. Phys Rev E, 2004, 69: 026113
42 Blondel V D, Guillaume J L, Lambiotte R, et al. Fast unfolding of communities in large networks. J Stat Mech: Theory Exp, 2008, 2008: P10008
43 Arab M, Afsharchi M. Community detection in social networks using hybrid merging of sub-communities. J Netw Comput Appl, 2014, 40: 73–84
44 Wakita K, Tsurumi T. Finding community structure in mega-scale social networks. In: Proceedings of the 16th International Conference on World Wide Web, Banff, 2007. 1275–1284
45 Ovelgönne M, Geyer-Schulz A, Stein M. Randomized greedy modularity optimization for group detection in huge social networks. In: Proceedings of the fourth SNA-KDD Workshop, Washington, 2010. 1–9
46 Shang R H, Bai J, Jiao L C, et al. Community detection based on modularity and an improved genetic algorithm. Phys A: Stat Mech Appl, 2013, 392: 1215–1231
47 Leicht E A, Newman M E J. Community structure in directed networks. Phys Rev Lett, 2008, 100: 118703
48 Ya W J, Cai Y J, Jian Y. An efficient community detection algorithm using greedy surprise maximization. J Phys A: Math Theor, 2014, 47: 165101
49 Li K, Pang Y. A unified community detection algorithm in complex network. Neurocomputing, 2014, 130: 36–43
50 Liu X, Yi D Y. Complex network community detection by local similarity. Acta Automatica Sinica, 2011, 37: 1520–1529 [刘旭, 易东云. 基于局部相似性的复杂网络社区发现方法. 自动化学报, 2011, 37: 1520–1529]
51 Lancichinetti A, Fortunato S, Radicchi F. Benchmark graphs for testing community detection algorithms. Phys Rev E, 2008, 78: 046110

Parallel overlapping community detection algorithm in complex networks based on label propagation

Chunying LI[1,2], Yong TANG[1*], Hai LIN[1], Chengzhe YUAN[1] & Huiqiang MAI[1]

1 *School of Computer, South China Normal University, Guangzhou* 510631*, China*;
2 *Computer Network Center, Guangdong Polytechnic Normal University, Guangzhou* 510665*, China*
*E-mail: ytang@scnu.edu.cn

Abstract Some of the defects of community detection algorithms in complex networks include pre-parameter limits and redundant labels. Accordingly, we present an improved overlapping community detection algorithm in complex networks that is a minimal maximal clique label propagation algorithm (MMCLPA). The algorithm uses the minimal maximal clique (MMC), and allows each MMC to share the same label by reducing redundant labels and random factors. Therefore, the MMCLPA algorithm increases the stability of the algorithm. Meanwhile, the MMCLPA algorithm completes label propagation from core node groups (MMC) for distribution using intimacy as the weight of the label. For the label's post processing, uses adaptive threshold method to overcome pre-parameter limitations in unknown complex networks. When compared with other community detection algorithms using artificial and real network data, our experiment results prove that the MMCLPA algorithm not only increases the tolerance for mixing parameters, but also improves the robustness of the algorithm. Lastly, by employing a distributed computing model (Map Reduce) and cloud platform (Hadoop), we allow parallelization implementation of MMCLPA. Our experiment results show that parallel MMCLPA owns an approximation of quality compared with the stand-alone system, as well as better scalability.

Keywords complex network, community detection, overlapping community, label propagation, parallel computing

Chunying LI was born in 1978. She received her M.S. degree from Beijing Institute of Technology in 2007, and she is pursuing her Ph.D. degree from the School of Computer Science, South China Normal University, Guangzhou City. Currently, she is an associate professor of the Computer Network Center, Guangdong Polytechnic Normal University. Her research interests include social network services and cooperative computing.

Yong TANG was born in 1964. He received his Ph.D. degree from the University of Science and Technology of China, Beijing, in 2001. Currently, he is a professor and dean of the School of Computer Science, South China Normal University. His research interests include temporal databases, cooperative computing, cloud computing, and social network services.