基于影响力与种子扩展的重叠社区发现

於志勇[1,2,3], 陈基杰[1,2,3],郭 昆[1,2,3],陈羽中[1,2,3],许 倩[4]
(1. 福州大学数学与计算机科学学院,福建福州 350116;2. 福建省网络计算与智能信息处理重点实验室,福建福州 350116;
3. 空间数据挖掘与信息共享教育部重点实验室,福建福州 350116;4. 国网信通亿力科技有限责任公司,福建福州 350003)

摘 要: 社区发现作为复杂社交网络中一个重要的研究方向.针对目前基于种子节点的算法在种子选取与扩展等方面的不足,提出了一种基于影响力与种子扩展的重叠社区发现算法(Influence Seeds Extension Overlapping Community Detection,简称 i-SEOCD 算法).首先,利用节点影响力策略找出具有紧密结构的种子社区.其次,从这些种子社区出发,计算社区邻居集节点与社区的相似度,并取出相似度超过设定阈值的节点.然后,采用优化自适应函数的策略来扩展社区.最后,对网络中的自由节点进行社区隶属划分,进而实现了整个网络的重叠社区结构挖掘.在真实社交网络和人工生成网络上实验表明,i-SEOCD 算法能够准确、快速地发现复杂网络中的重叠社区结构.

关键词: 局部社区发现; 种子扩展; 节点影响力; 重叠社区
中图分类号: TP393 **文献标识码:** A **文章编号:** 0372-2112 (2019)01-0153-08
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2019.01.020

Overlapping Community Detection Based on Influence and Seeds Extension

YU Zhi-yong[1,2,3], CHEN Ji-jie[1,2,3], GUO Kun[1,2,3], CHEN Yu-zhong[1,2,3], XU Qian[4]
(1. *College of Mathematics and Computer Sciences, Fuzhou University, Fuzhou, Fujian* 350116, *China*;
2. *Fujian Provincial Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou, Fujian* 350116, *China*;
3. *Ministry of Education Key Laboratory of Spatial Data Mining & Information Sharing, Fuzhou, Fujian* 350116, *China*;
4. *State Grid Info-Telecom Great Power Science and Technology Co. Ltd., Fuzhou, Fujian* 350003, *China*)

Abstract: Community detection is a significant research direction in the research of social networks. To improve the quality of seeds selection and expansion, we propose an influence seeds extension overlapping community detection (i-SEOCD) algorithm for overlapping community detection. First, i-SEOCD uses a node influence strategy to find the seed communities with tight structures. Second, on the basis of the seed communities, we calculate the similarity among communities and their neighbor nodes. The nodes whose similarity is greater than a predefined threshold are selected. Third, the strategy of optimizing a self-adaptive function is adopted to expand the communities. Finally, the free nodes in the network are assigned to their corresponding communities in order to find out all the overlapping community structures. Experiments on the real and artificial networks show that i-SEOCD is capable of discovering overlapping communities in complex social networks efficiently.

Key words: local community detection; seeds extension; node influence; overlapping community

1 引言

随着社会信息网络的快速发展,出现了很多复杂的网络结构,例如社交网络、科学家合作网络、蛋白质互相协作网等[1].复杂网络一般用图结构来表示,节点代表网络中的个体,边代表个体间的联系[2].复杂网络中

收稿日期:2018-01-29;修回日期:2018-07-11;责任编辑:李勇锋
基金项目:国家自然科学基金(No. 61300104, No. 61772136, No. 61672158);福建省高校杰出青年科学基金(No. JA12016);福建省高等学校新世纪优秀人才支持计划(No. JA13021);福建省杰出青年科学基金(No. 2014J06017, No. 2015J06014);福建省科技创新平台计划(No. 2009J1007, No. 2014H2005);福建省自然科学基金(No. 2013J01230, No. 2014J01232);福建省高校产学合作项目(No. 2014H6014, No. 2017H6008);海西政务大数据应用协同创新中心

的社区结构通常表现为社区内的点连接紧密,而社区间的点连接稀疏. 社区发现就是研究复杂网络结构的关键技术之一[3]. 目前,社区发现的研究成果可以被应用于网络舆情监控、个性化兴趣推荐、蛋白质功能预测等诸多领域.

为了挖掘复杂网络的社区结构,近些年已有很多人对其展开深入研究. 传统的社区发现算法包括层次聚类算法[4]、谱方法[5]、基于团的方法[6]、边聚类[7]、标签传播[8]等. 这些算法虽然可以较好发现网络社区结构,但要知道整个网络结构的信息,当网络规模较大或者不完整时就会受到一些约束.

基于种子扩展的局部社区发现算法通常是从网络中的种子节点出发,利用社区的局部信息来发现网络的社区结构. 2009 年,Lancichinetti 等人[9]提出了基于局部扩展的重叠社区发现算法 LFM,通过判断节点能否增加社区的自适应函数 fitness 值来决定节点是否加入社区;Coscia 等人[10]提出的 DEMON 算法是以整个网络所有节点作为起始点,并通过邻域信息来扩展社区;Su 等人[11]基于随机游走的策略来把节点加入最可能属于的社区中. 由于基于种子节点扩展的局部社区发现算法都是从种子开始扩展,种子节点的好坏直接影响了社区发现的质量. Chen 等人[12]以网络中度最大的节点作为种子来扩展社区;Whang 等人[13]通过聚类中心思想和最大度原则来选择种子;Cravino 等人[14]基于局部影响的评估函数来选择种子,并使用静态方法发现社区结构;Wang 等人[15]提出了结构中心性节点概念,并基于此进行局部社区扩展,该方法精度高但只适用于规模较小的网络.

针对种子选取策略和社区扩展策略的一些不足,本文提出了一种基于影响力与种子扩展的重叠社区发现算法(i-SEOCD). 在种子选取阶段,通过计算每个节点的影响力分数,选出核心种子节点,然后利用相似度结合节点的邻域信息构成初始种子社区;在种子扩展阶段,引入节点与社区的相似度计算,然后再对自适应函数进行优化来扩展社区,加强了社区扩展的有效性与稳定性;对网络中自由节点进行处理,i-SEOCD 算法较传统算法提高了社区划分的准确性;能够启发式的检测社区重叠节点,进而可以发现网络中重叠的社区结构.

2 社区发现相关概念

网络可以用无向图 $G=(V,E)$来表示,其中,V 表示节点集合,E 表示边集合. 下面将对本文所使用的基本概念进行描述.

定义 1(社区邻居集) 社区邻居集 $N_s(C)$表示与社区 C 有直接连接边的节点集合:

$$N_s(C)=\bigcup\nolimits_{v\in C}\Gamma(v)-C \tag{1}$$

$$\Gamma(v)=\{u:u\in V,(v,u)\in E\} \tag{2}$$

其中,式(1)中 C 代表一个社区,式(2)为节点 v 的邻居集合 $\Gamma(v)$的定义.

定义 2(Jaccard 系数[16]) 两个节点之间的 Jaccard 系数 J_{uv}公式定义为:

$$J_{uv}=\frac{|\Gamma(v)\cap\Gamma(u)|}{|\Gamma(v)\cup\Gamma(u)|} \tag{3}$$

Jaccard 系数 J_{uv}可以用来衡量节点之间的亲密度,J_{uv}值越大代表两个节点之间越相似.

定义 3(影响力分数) 节点 v 的影响力分数 $I_{\text{score}}(v)$定义为:

$$I_{\text{score}}(v)=k_v\times\sum_{u\in\Gamma(v)}(k_u\times J_{uv}) \tag{4}$$

其中,k_v为节点 v 的度. $I_{\text{score}}(v)$越大,则代表节点 v 在网络中所具有的影响力越大.

定义 4(节点与社区相似度) 节点 v 与社区 C 的相似度 $S_{nc}(v,C)$定义为:

$$S_{nc}(v,C)=\frac{|N_s(C)\cap\Gamma(v)|}{|N_s(C)\cup\Gamma(v)|} \tag{5}$$

$S_{nc}(v,C)$越大,表明节点 v 属于社区 C 的概率越大.

定义 5(社区相似度) 社区 C_i与社区 C_j的相似度 $S_{cc}(C_i,C_j)$定义为:

$$S_{cc}(C_i,C_j)=\frac{|C_i\cap C_j|}{\min(|C_i|,|C_j|)} \tag{6}$$

$S_{cc}(C_i,C_j)$越大,则表明社区 C_i与社区 C_j的结构越相近. 一般当 $S_{cc}(C_i,C_j)$大于 0.5 时,就可以将两个社区合并为一个社区.

定义 6(自适应函数) 自适应 fitness 函数用于衡量一组节点的紧密程度,具体公式定义如下:

$$f_g(C)=\frac{k_{\text{in}}^g}{(k_{\text{in}}^g+k_{\text{out}}^g)^\alpha} \tag{7}$$

其中,k_{in}^g和 $k_{\text{in}}^{\text{out}}$分别为社区 C 内部度的总值和外部度的总值,参数 α 是一个为正的实数,用来控制发现的社区规模.

3 i-SEOCD 算法设计与实现

3.1 算法总体设计思想

基于影响力与种子扩展的局部重叠社区发现算法主要由 4 个阶段组成:1)种子社区检测;2)相似种子社区合并;3)社区扩展挖掘;4)社区优化. 种子社区的检测主要是通过计算每个节点的影响力分数,然后根据邻域的局部信息选出核心种子节点,并与具有紧密结构的邻居节点构成种子社区;在社区扩展阶段利用局部节点及邻居节点相关信息,选取与社区具有较高相似度且能够优化 fitness 函数的节点加入社区,以此来实现整个网络的社区划分. 由于每个种子社区都是沿着其邻居集独立进行社区扩展,因此可以发现网络中的

重叠结构.

3.2 种子社区检测

在网络中,核心种子节点选取的好坏直接影响了局部社区发现的准确性. 受 Cravino 等人[14]提出的节点影响力的启发,设计了一种新的种子选取过程. 首先采用 Jaccard 系数结合式(4)计算出每个节点 v 的影响力分数 $I_{\text{score}}(v)$;然后统计每个节点 v 的分数大于其邻居节点分数的个数 l_{num},若 l_{num} 与节点 v 邻居节点个数 n_{num} 的比值大于阈值 ρ,则将节点 v 定义为核心种子节点;接着使用式(5)找出节点 v 邻居中与初始种子社区的相似度 S_{nc} 大于阈值 ε 的节点,加入到初始种子社区中得到最后的初始种子社区 S. 具体步骤如函数 1 所示.

函数 1　detectSeedsCommunity

输入:网络 $G(V,E)$,ε,ρ
输出:种子社区集合 *Seeds*
1. *Seeds* = Φ;S_{score} = 0;
2. FOR EACH $v \in V$
3.　$S_{\text{score}}(v) = I_{\text{score}}(v)$;　//根据式(4)计算节点 v 的评分
4. END FOR
5. Li_{core} = Φ;　//初始化核心节点列表
6. FOR EACH $v \in V$
7.　$l_{\text{num}} = 0$; $n_{\text{num}} = |\varepsilon(v)|$;
8.　FOR EACH $z \in \varepsilon(v)$
9.　　IF $S_{\text{score}}(v) > S_{\text{score}}(z)$ THEN $l_{\text{num}} += 1$;
10.　END FOR
11.　IF ($l_{\text{num}}/n_{\text{num}} > \rho$) THEN $\text{Li}_{\text{core}} = \text{Li}_{\text{core}} \cup v$;
12. END FOR
13. FOR EACH $v \in \text{Li}_{\text{core}}$
14.　IF $v \notin Seeds$ THEN
15.　　C_s = Φ;　$C_s = C_s \cup v$;
16.　　FOR EACH $z \in \varepsilon(v)$
17.　　$sim = S_{nc}(z, C_s)$;　//根据式(5)计算
18.　　IF $sim > \varepsilon$ THEN $C_s = C_s \cup z$;
19.　　END FOR
20.　　$Seeds = Seeds \cup C_s$;
21.　END IF
22. END FOR

3.3 相似种子社区合并

在检测种子社区阶段,可能会出现两个种子社区的很相似的情况,因此需要将其合并,避免后面种子扩展阶段不必要的重复计算. 根据式(6)计算社区之间的相似度 $S_{cc}(C_i, C_j)$,若 $S_{cc}(C_i, C_j)$ 大于阈值 ε,则将两个种子社区合并,从而得到更加稳定紧密的种子社区集合 *Seeds′*. 具体步骤如函数 2 所示.

函数 2　mergeSimilarSeeds

输入:种子社区集合 *Seeds*,ε
输出:合并后的种子社区集合 *Seeds′*
1. *Seeds′* = Φ;　//初始化新种子集
2. FOR EACH $s \in Seeds$
3.　IF *Seeds′* = Φ THEN $Seeds' = Seeds' \cup s$;
4.　*condition* = True;//判定条件
5.　FOR EACH $s' \in Seeds'$
6.　　$sim = S_{cc}(s, s')$;　//根据式(6)计算
7.　　IF $sim > \varepsilon$ THEN
8.　　$s_{merge} = s \cup s'$;　//合并种子社区 s 与种子社区 s'
9.　　$Seeds' = Seeds' \cup s_{merge}$;$Seeds' = Seeds' - s'$;
10.　　*condition* = False;
11.　　BREAK;
12.　END IF
13.　END FOR
14.　IF *condition* THEN $Seeds' = Seeds' \cup s$;
15. END FOR

3.4 社区扩展挖掘

得到稳定紧密的种子社区之后,就可以进行社区扩展挖掘. 扩展思路如下:首先获取种子社区的邻居集 N_s,根据式(5)计算 N_s 中每个邻居节点与社区的相似度 S_{nc},选出相似度大于阈值 ε 的节点作为候选节点. 然后计算这些候选节点加入局部社区后的 fitness 函数值,把能够让 fitness 的函数值增加的候选节点加入社区,否则将其置为网络中的自由节点,同时删除社区中 fitness 函数增量为负的节点. 最后更新 N_s 并继续重复以上步骤,直到 N_s 为空. 具体步骤如函数 3 所示.

函数 3　communityExtendMining

输入:网络 G,种子集 *Seeds′*,ε
输出:重叠社区集合 C
1. C = Φ;　//初始化重叠社区集合
2. FOR EACH $s \in Seeds'$
3.　$C_s = s$;　//初始化社区
4.　calculate $N_s(C_s)$;　//根据式(1)计算社区邻居集
5.　WHILE $N_s(C_s)$! = Φ
6.　　$\text{Li}_{\text{candidate}}$ = Φ;　//初始化候选节点列表
7.　　FOR EACH $z \in N_s(C_s)$
8.　　$sim = S_{nc}(z, C_s)$;　//根据式(5)计算
9.　　IF $sim > \varepsilon$ THEN $\text{Li}_{\text{candidate}} = \text{Li}_{\text{candidate}} \cup z$;
10.　END FOR
11.　FOR EACH $v \in \text{Li}_{\text{candidate}}$
12.　　$f_g^+ = f_g(C_s \cup v)$;　//根据式(7)计算
13.　　IF $f_g^+ > f_g(C_s)$ THEN
14.　　$C_s = C_s \cup v$;
15.　　FOR EACH $z \in C_s$
16.　　$f_g^- = f_g(C_s - z)$;
17.　　IF $f_g^- > f_g(C_s)$ THEN $C_s = C_s - z$;
18.　　END FOR
19.　　END IF
20.　　END FOR
21.　　*recalculate*$N_s(C_s)$;　//重新计算社区邻居集

22. END WHILE
23. $C = C \cup C_s$;
24. END FOR

3.5 社区优化

在社区扩展过程中,网络中可能还会存在未属于任何社区的自由节点,而且社区集合中还会出现相似度高的社区. 因此,需要对社区进行优化,即对自由节点进行社区分配或让其独立形成一个社区,然后合并相似度较高的社区. 优化主要分为两个步骤:第一步,计算节点与各个社区的相似度 S_{nc},当 S_{nc} 大于阈值 ε 时就把节点加入该社区,否则就让其形成一个单独社区;第二步,计算社区与社区之间的相似度 S_{cc},当 S_{cc} 大于阈值 ε 时,将社区进行合并. 最后得到网络社区划分的结果.

3.6 算法复杂度分析

设网络有 n 个节点和 m 条边,网络的平均度为 k. 函数 1 中,计算每个节点影响力分数的时间复杂度为 $O(k^2 n)$,生成紧密种子社区的时间复杂度为 $O(kp)$,p 为种子社区数目. 则函数 1 总的时间复杂度为 $O(k^2 n + kp)$. 函数 2 中,合并相似种子社区时间复杂度为 $O(p^2)$. 函数 3 需要根据种子社区邻居集迭代的进行社区扩展挖掘,其时间复杂度为 $O(pnk\log k + pnk\log n)$. 社区优化中,处理自由节点社区分配的时间复杂度为 $O(pq)$,q 为自由节点数,合并相似社区的时间复杂度为 $O(p^2)$. 则社区优化的总时间复杂度为 $O(pq + p^2)$. 综合上述分析,由于在一般社交网络中,$k \ll n$,i-SEOCD 算法的总时间复杂度为 $O(n + n\log n + pq + p^2)$. 通过控制算法参数 ρ 可以使 $p \ll n$. 在实际网络中,自由节点的数量 q 一般较少. 因此,i-SEOCD 算法的总时间复杂度可简化为 $O(n\log n)$,算法具有近似线性的时间复杂度.

接下来分析算法的空间复杂度. 函数 1 需要存储整个网络的影响力分数和 Jaccard 系数,因此占用空间为 $O(n+m)$. 函数 2 合并相似种子社区,需要的空间为 $O(n)$. 函数 3 扩展种子社区时需要存储每个种子的邻居集,因此空间消耗为 $O(np)$. 函数 4 处理自由节点和合并相似社区需要的空间为 $O(n)$. 综合上述分析,i-SEOCD 算法的空间复杂度为 $O(n + m + n + np + n)$,由于在一般社交网络中 $p \ll n$,因此,算法总的空间复杂度为 $O(m)$.

4 实验结果与分析

为了检测 i-SEOCD 算法的性能,分别采用了真实网络和不同规模的人工数据集网络进行对比实验. 实验环境为:一台配置为 3.1GHz Pentium 4CPU,16G 内存的 PC 机,操作系统为 Windows7 64 位. 所有算法代码均是基于 Python 2 实现.

4.1 实验数据集

(1)真实数据集

采用的真实社交网络包括 Zachary 空手道俱乐部网络 Karate[17]、海豚关系网 Dolphins[18]、美国大学足球赛网络 Football[4]、美国政治书籍网 Polbooks[19]、爵士乐音乐家关系网 Jazz[20]、科学合作网 Ca-GrQc[21] 和 Ca-HepTh[21].

(2)人工数据集

LFR-benchmark 基准程序[22]是近年来较为广泛使用的人工基准网络程序,因为其生成的网络可以很好的表示出节点度和社区规模分布的异质性. 利用 LFR 基准程序共生成了 5 组人工模拟网络,基本参数设置分别为:

1) $D1$: $N = 1000$, $\mu = 0.1 \sim 0.5$, $on = 0.1$, $om = 3$;

2) $D2$: $N = 5000$, $\mu = 0.1 \sim 0.7$;

3) $D3$: $N = 10000 \sim 200000$, $\mu = 0.3, 0.6$;

4) $D4$: $N = 5000$, $\mu = 0.3$, $on = 0.1, 0.3$, $om = 2 \sim 10$;

5) $D5$: $N = 10000 \sim 200000$, $\mu = 0.1$;

其余参数均采用相同的设置:$k = 20$, $k_{\max} = 50$, $C_{\min} = 20$, $C_{\max} = 100$.

4.2 实验方法和评价指标

(1)实验方法

为了比较 i-SEOCD 算法性能的好坏,把该算法与多个重叠社区发现算法作比较,对比的算法包括 LFM 算法[9]、CPM 算法[6]、DEMON 算法[10]、SLPA 算法[8]、NISE 算法[13]、LCCDO 算法[15]. 将这些算法在真实网络和人工生成网络数据集上进行对比分析,以此来评价 i-SEOCD 算法的准确性.

(2)评价指标

采用沈华伟等人[23]提出的评估重叠社区结构质量的模块度 EQ 来作为真实网络划分的评价指标. EQ 值越接近 1,则表示网络划分出的社区质量越好. 模块度 EQ 的定义如下:

$$\mathrm{EQ} = \frac{1}{2m}\sum_{k=1}^{c}\sum_{i,j\in C_k}\frac{1}{O_i O_j}\left[A_{ij} - \frac{k_i k_j}{2m}\right] \tag{8}$$

其中,m 为网络中边的总数;c 为划分得到的社区的数目;O_i 为节点 i 所属的社区个数;k_i 为节点 i 的度;A_{ij} 用于判断节点 i 和节点 j 之间是否存在连接,若存在连接则 A_{ij} 为 1,否则为 0.

采用标准化互信息[24](Normalized Mutual Information,简称 NMI)作为人工生成网络的评价指标. 其公式定义如下:

$$\mathrm{NMI} = \frac{-2\sum_{i=1}^{C_A}\sum_{j=1}^{C_B} N_{ij}\log\left(\frac{N_{ij}\times N}{N_{i\cdot}\times N_{\cdot j}}\right)}{\sum_{i=1}^{C_A} N_{i\cdot}\log\left(\frac{N_{i\cdot}}{N}\right) + \sum_{j=1}^{C_B} N_{\cdot j}\log\left(\frac{N_{\cdot j}}{N}\right)} \tag{9}$$

其中,C_A 为标准社区划分的结果,C_B 为算法所得到社区

划分的结果,矩阵 N 的行对应标准社区结果,列对应算法得到的社区检测结果,$N_{i.}$ 为第 i 行的总和,$N_{.j}$ 为第 j 列的总和. NMI 值越大,说明算法划分社区的效果越好.

4.3　参数实验

对于 i-SEOCD 算法,主要用到了三个参数,分别为 α、ε、ρ.

(1)参数 α

参数 α 是用来控制社区发展的规模,一般为正实数. 参数 α 与 LFM 算法中的取值范围一样,设置为 0.8 ~1.2.

(2)参数 ε

参数 ε 是用来衡量节点与社区相似度的阈值,若两个节点越相似,其相似度越接近 1;反之,越接近 0. 因此将参数 ε 控制在 0 到 1 内进行实验.

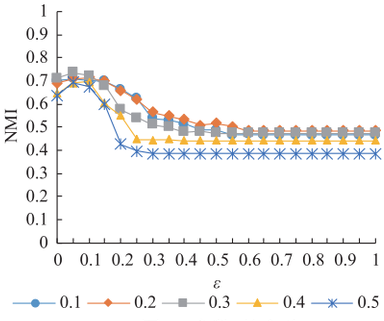

图1　参数ε的实验

图 1 为 i-SEOCD 算法在 D1 组人工模拟网络上控制参数 ε 从 0 到 1 变化的实验结果. D1 组模拟网络共有 5 套数据集,LFR 参数 μ 值从 0.1 ~0.5 变化. 从图中可以看出,多个数据集 NMI 值的曲线变化形式几乎都一样,因此参数 ε 并不受数据集的影响. 当 ε 取 0.05 时效果最好,随着 ε 逐渐增大,NMI 值降低,最终在 0.3 ~1 范围内保持不变. 经过实验结果分析,i-SEOCD 算法中参数 ε 的取值范围在 0 ~0.1 之间比较合适.

(3)参数 ρ

参数 ρ 是用来寻找核心种子节点的阈值参数. 因为参数 ρ 为节点 v 影响力分数大于邻居节点分数个数 l_{num} 与邻居节点总数 n_{num} 的比值,而 l_{num} 的取值为 0 ~ n_{num},因此将参数 ρ 控制在 0 到 1 变化进行参数实验.

图 2 为 i-SEOCD 算法在 D1 组人工模拟网络上控制参数 ρ 从 0 到 1 变化的实验结果. 从图中可以看出,随着 ρ 发生变化,多个数据集的 NMI 值曲线走势也接近一样,因此可知参数 ρ 的取值也不受数据集变化的影响. 当参数 ρ 取 0 到 0.9 之间时 NMI 结果变化波动不大,取 1 时 NMI 结果最差. 因为参数 ρ 是用来选取核心种子节点的阈值,ρ 取值较小时,就会有更多的节点成为种子节点,如果设置较大时,又很难找到有效的种子节点,从而选取的种子质量降低,导致算法的精度不高.

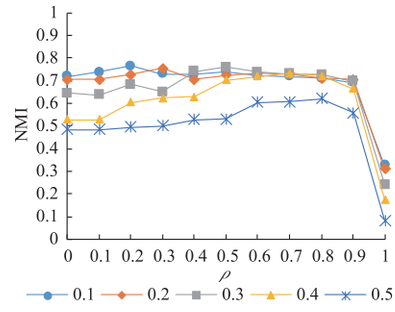

图2　参数ρ的实验

从实验结果中可以得出,i-SEOCD 算法中参数 ρ 的取值范围在 0.8 ~0.9 之间比较合适.

4.4　真实数据集上的实验结果

图 3 为 i-SEOCD 算法与其他 6 种社区发现算法在真实数据集中的实验结果. 从图中可以看出,i-SEOCD 算法除了在 Jazz 和 Ca-HepTh 数据集上模块度比 SLPA 算法略低一点,在其它真实网络上的模块度均高于其他算法,表明 i-SEOCD 算法具有较好的性能. 因为 i-SEOCD 采用影响力分数的方式选取高质量的种子社区,从而可以有效的发现真实网络中的社区结构.

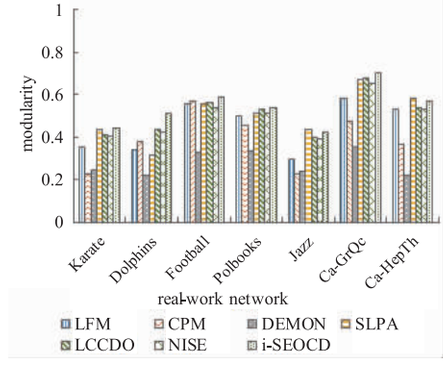

图3　真实数据集上的精度实验结果

4.5　人工数据集上的实验结果

(1)不同 μ 值的实验结果

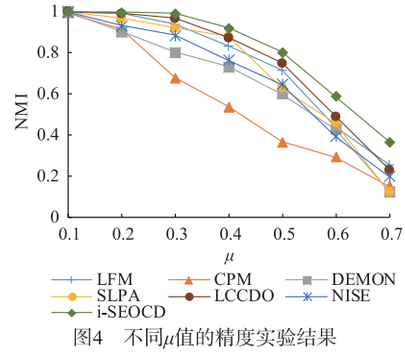

图4　不同μ值的精度实验结果

图 4 为不同算法在 D2 组人工模拟网络上的实验

结果. 从图中可以看出尽管 μ 取不同的值,i-SEOCD 算法 NMI 值均高于其他算法,且随着 μ 的增大,NMI 的值下降的速度也比其他算法慢,反映出 i-SEOCD 算法采用影响力分数方式对种子选取和扩展策略的有效性,同时验证了 i-SEOCD 算法对参数进行实验是有必要的. 其中,CPM 算法将社区看作团,若网络中团结构不明显就算法精度造成影响;DEMON 算法需要对整个网络的节点进行社区扩展,容易形成多个独立社区;LFM 和 SLPA 算法都是随机选择网络中的节点为初始点,具有较强的随机性;NISE 算法采用节点的所有邻居集作来进行社区扩展,所以很容易形成大的社区,对算法精度造成影响;LCCDO 算法没有处理网络中的自由节点,因此精度会略低一点.

(2)不同规模数据集的实验结果

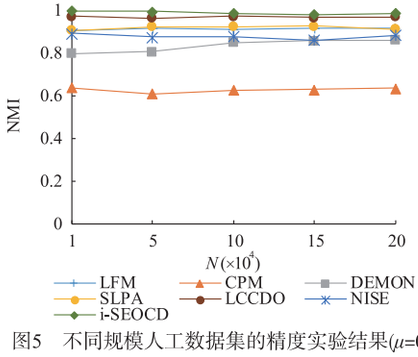

图5 不同规模人工数据集的精度实验结果(μ=0.3)

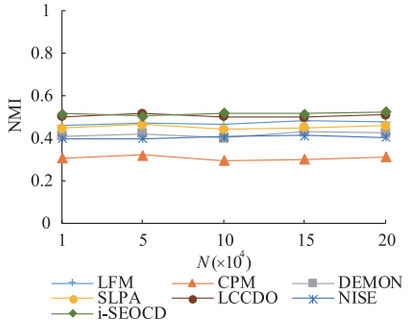

图6 不同规模人工数据集的精度实验结果(μ=0.6)

图 5、图 6 为不同算法在 D3 组人工模拟网络上的实验结果. 从图中可以分析出,尽管 μ 取不同值,随着网络规模逐渐增大,i-SEOCD 算法的 NMI 值基本保持不变,且均比其他算法高. 因为 i-SEOCD 算法在种子寻找和社区扩展阶段都考虑了节点与社区相似度的计算,选取有效的节点加入社区,充分利用网络节点的局部信息来进行网络的社区划分. i-SEOCD 算法还对网络的中自由节点做处理,将节点划分到相似社区中或自己形成一个独立社区,提高了算法发现社区的质量. 所以,在不同规模及不同社区结构的网络中,i-SEOCD 算法始终具有较高且平稳的社区检测准确率.

(3)不同 *om* 值的实验结果

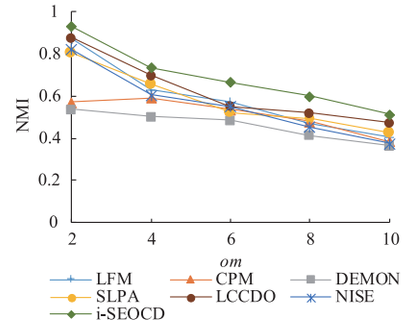

图7 不同*om*精度实验结果(*on*=0.1)

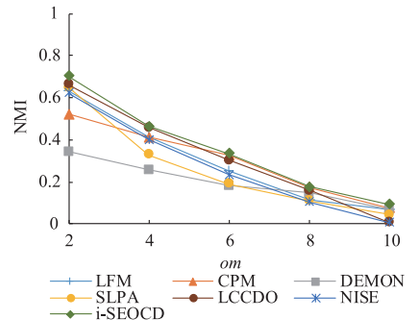

图8 不同*om*精度实验结果(*on*=0.3)

图 7、图 8 为各个算法在 D4 组人工模拟网络上的实验结果. 从图中可以看出,尽管 *on* 取不同值,i-SEOCD 算法的 NMI 值高于其它算法. 随着网络重叠度的提高,网络结构变复杂,因此所有算法精度都会下降. 由于 i-SEOCD 算法在用影响力方式得到紧密的种子社区之后,对相似度较高的种子社区进行合并,从而提高了发现高质量的社区的能力. 接着再利用相似度以及优化 fitness 函数扩展不同的种子社区,启发式的发现网络中的重叠节点. 因此,i-SEOCD 算法可以有效的发现网络中的重叠结构.

(4)运行时间实验结果

图 9 为为 i-SEOCD 算法和其他 6 种算法在 D5 组人工模拟网络上运行时间结果. 从图中可以看出,除了 SLPA 算法和 NISE 算法,i-SEOCD 算法的时间效率都优于其他的算法. 由于 i-SEOCD 算法在种子社区检测和社区扩展阶段做了改进,加入影响力的计算和相似度的条件判断,提高了算法的时间效率. 其中 SLPA 算法基于标签传播的策略进行社区发现,因此时间效率较高;NISE 算法采用 PageRank 策略进行社区扩展,比较适合处理大规模的网络;DEMON 算法需要对网络中所有节点进行社区扩展,最后还需对挖掘出的社区进行优化,因此时间效率很低,在节点数为 15 万规模时,DEMON 算法的运行时间超过了 2h,因此其后续的结果

值未在图中画出;LCCDO 算法在确定网络中心结构时需要计算节点间的最短路径,因此时间效率也较低;CPM 算法的时间复杂度为非多项式级;LFM 算法是随机选择种子节点进行扩展,扩展过程考虑是整个邻域信息,因此时间开销也相对较大. 综合考虑社区划分精度和时间运行效率,i-SEOCD 算法性能较好,而且能够有效的处理大规模网络数据.

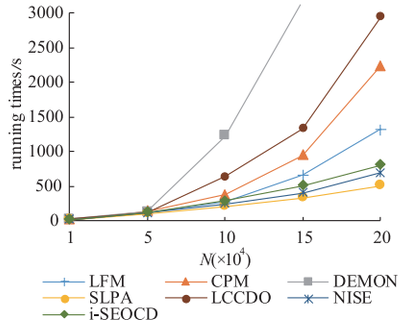

图9　算法运行时间对比

5　总结与展望

本文提出了一种基于影响力与种子扩展的重叠社区发现算法(i-SEOCD),用于发现网络的社区结构. 首先,计算出每个节点影响力分数,找出网络中最具有影响力的节点作为核心种子节点,并与其紧密连接的邻居节点一起构成种子社区. 其次,合并相似的种子社区,从而减少一些重复计算. 然后,计算种子社区邻居集的节点与社区的相似度,采用 fitness 函数来扩展社区. 最后,优化社区,把网络中未属于任何社区的节点加入相似度最大的社区,并合并相似度高的社区,提高社区发现质量. 在真实和人工数据集上的实验表明:i-SEOCD 算法在具备近似线性时间复杂度的同时,能够准确地检测出社区节点,并可以很好划分重叠社区. 未来的工作将考虑把 i-SEOCD 算法与更多新的算法做比较分析. 然后考虑采用层次结构信息来改进算法的社区扩展部分,使其能发现网络中带有层次的社区. 同时,为了适应网络的动态变化,将引入动态社区发现策略,以实现动态演化的社区结构挖掘,进一步增强算法的实用性.

参考文献

[1] Newman M E. The structure of scientific collaboration networks[J]. Proceedings of the National Academy of Sciences of the United States of America,2001,98(2):404-409.

[2] Fortunato S. Community detection in graphs[J]. Physics Reports,2012,486(3):75-174.

[3] Radicchi F,Castellano C,Cecconi F,et al. Defining and identifying communities in networks[J]. Proceedings of the National Academy of Sciences of the United States of America,2003,101(9):2658.

[4] Girvan M,Newman M E. Community structure in social and biological networks[J]. Proceedings of the National Academy of Sciences,2002,99(12):7821-7826.

[5] Luxburg,Ulrike. A tutorial on spectral clustering[J]. Statistics & Computing,2007,17(4):395-416.

[6] Palla G,Derényi I,Farkas I,et al. Uncovering the overlapping community structure of complex networks in nature and society[J]. Nature,2005,435(7043):814.

[7] Kim Y,Jeong H. Map equation for link communities[J]. Physical Review E,2011,84(2):026110.

[8] Xie J,Szymanski B K. Towards linear time overlapping community detection in social networks[A]. Pacific-Asia Conference on Knowledge Discovery and Data Mining[C]. Springer,Berlin,Heidelberg,2012. 25-36.

[9] Lancichinetti A,Fortunato S,Kertész J. Detecting the overlapping and hierarchical community structure in complex networks[J]. New Journal of Physics,2009,11(3):033015.

[10] Coscia M,Rossetti G,Giannotti F,et al. Demon:a local-first discovery method for overlapping communities[A]. Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining[C]. ACM,2012. 615-623.

[11] Su Y,Wang B,Zhang X. A seed-expanding method based on random walks for community detection in networks with ambiguous community structures[J]. Scientific Reports,2017,7:41830.

[12] Chen Q,Wu T T,Fang M. Detecting local community structures in complex networks based on local degree central nodes[J]. Physica A:Statistical Mechanics and Its Applications,2013,392(3):529-537.

[13] Whang J J,Gleich D F,Dhillon I S. Overlapping community detection using neighborhood-inflated seed expansion[J]. IEEE Transactions on Knowledge and Data Engineering,2016,28(5):1272-1284.

[14] Cravino N,Figueira Á. Community detection by local influence[A]. Advances in Information Systems and Technologies[M]. Springer,Berlin,Heidelberg,2013. 193-200.

[15] Wang X,Liu G,Li J. Overlapping community detection based on structural centrality in complex networks[J]. IEEE Access,2017,5:25258-25269.

[16] Jaccard P. Etude de la distribution florale dans une portion des Alpes et du Jura[J]. Bulletin De La Societe Vaudoise Des Sciences Naturelles,1901,37(142):547-579.

[17] Zachary W W. An information flow model for conflict and

fission in small groups[J]. Journal of anthropological research,1977,33(4):452 – 473.
[18] Lusseau D, Schneider K, Boisseau O J, et al. The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations[J]. Behavioral Ecology and Sociobiology,2003,54(4):396 – 405.
[19] Newman M E J, Girvan M. Finding and evaluating community structure in networks[J]. Physical review E,2004,69(2):026113.
[20] Gleiser P M, Danon L. Community structure in jazz[J]. Advances in Complex Systems,2003,6(04):565 – 573.
[21] Leskovec J, Kleinberg J, Faloutsos C. Graph evolution: densification and shrinking diameters[J]. ACM Transactions on Knowledge Discovery from Data (TKDD),2007,1(1):2.
[22] Lancichinetti A, Fortunato S, Radicchi F. Benchmark graphs for testing community detection algorithms[J]. Physical Review E,2008,78(4):046110.
[23] Shen H, Cheng X, Cai K, et al. Detect overlapping and hierarchical community structure in networks[J]. Physica A: Statistical Mechanics and Its Applications, 2009, 388(8):1706 – 1712.
[24] Danon L, Diaz-Guilera A, Duch J, et al. Comparing community structure identification[J]. Journal of Statistical Mechanics: Theory and Experiment,2005,2005(09):P09008.

作者简介

於志勇 男,1982 年生于湖北黄梅. 现为福州大学数学与计算机科学学院副教授. 研究方向为移动社会网络、普适计算.
E-mail:yuzhiyong@ fzu. edu. cn

陈基杰 男,1993 年生于福建三明. 现为福州大学数学与计算机科学学院研究生. 研究方向为数据挖掘、社交网络.
E-mail:jackie_cute@ 163. com

郭 昆 男,1979 年生于福建福州. 现为福州大学数学与计算机科学学院副教授. 研究方向为数据挖掘、灰色系统理论. 通信作者.
E-mail:gukn123@ 163. com

陈羽中 男,1979 年生于福建福州. 现为福州大学数学与计算机科学学院教授. 研究方向为云计算虚拟化技术、网络信息安全.
E-mail:yzchen@ fzu. edu. cn

许 倩 女,1984 年生于福建惠安. 现为国网信通亿力科技有限责任公司业务咨询师. 研究方向为大数据挖掘、通信、软件工程.
E-mail:150332181@ qq. com