Local Community Extraction for Non-overlapping and Overlapping Community Detection

Zhan Bu[1,⋆], Guangliang Gao[2], Zhiang Wu[1], Jie Cao[1], and Xiao Zheng[3]

[1] Jiangsu Provincial Key Laboratory of E-Business,
Nanjing University of Finance and Economics, Nanjing, China
[2] School of Computer Science and Engineering,
Nanjing University of Science and Technology, Nanjing, China
[3] School of computer science and technology,
Anhui University of Technology, Maanshan, China
buzhan@nuaa.edu.cn

Abstract. The scale of current networked system is becoming increasingly large, which exerts significant challenges to acquire the knowledge of the entire graph structure, and most global community detection methods often suffer from the computational inefficiency. Local community detection aims at finding a community structure starting from a seed vertex without global information. In this article, we propose a Local Community Extraction algorithm (LCE) to find the local community from a seed vertex. First, a local search model is carefully designed to determine candidate vertices to be preserved or discarded, which only relies on the local/incomplete knowledge rather than the global view of the network. Second, we expand LCE for the global non-overlapping community detection, in which the labels of detected local communities are seen as vertices' attributive tags. Finally, we adopt the results of LCE to calculate a membership matrix, which can been used to detect the global overlapping community of a graph. Experimental results on four real-life networks demonstrate the advantage of LCE over the existing degree-based and similarity-based local community detection methods by either effectiveness or efficiency validity.

Keywords: Complex Network, Incomplete Knowledge, Local Community Extraction, Non-overlapping Community, Overlapping community.

1 Introduction

A complex network is composed by a large number of highly interconnected dynamical nodes. Social , biological, and computer science networks are only a few examples of complex networks, and they often display a common topological feature-community structure. Discovering the latent communities therein is

⋆ Corresponding author.

X. Luo, J.X. Yu, and Z. Li (Eds.): ADMA 2014, LNAI 8933, pp. 98–111, 2014.

a useful way to infer some important functions. e.g., in social network, communities can be defined as subgroups whose members are "friends" to each other. A common formulation of the problem of community detection is to find a partitioning, $\mathcal{P} = \{\mathcal{C}_1, \cdots, \mathcal{C}_K\}$, of disjoint or joint subsets of vertices of the graph representing the network, in a meaningful manner.

In general, a community should be thought of a set of nodes that has more and/or better-connected edges between its members than between its members and the remainder of the network. The existing community definitions in the literature can be roughly divided into three categories, one is global-based [3,4,15], the other based on the vertex-similarity [8,17], and the third is local-based [13,16]. 1) The global-based definitions consider the graph as a whole, they follow the assumption that a graph has community structure if it is different from a random graph, e.g., the null model of Newman and Girvan [15]. 2) The vertex-similarity-based definitions are based the assumption that communities are groups of vertices similar to each other. e.g., the graph vertices can be embed in an n-dimensional Euclidean space, then the similarity of any vertex pair can be measured by their Euclidean distance. 3) The local-based definitions compare the internal and external cohesion of a sub-graph. The first recipe of this kind is LS-set [13], which stems from social network analysis. Some other local-based definitions can be found in recent literature.

Existing global-based and the vertex-similarity-based community detection approaches require clear pictures of the entire graph structure. And they are often described as global community detection(in short as GCD henceforth). As the scale of current networked system is becoming increasingly large, which exerts significant challenges to acquire the knowledge of the entire graph structure, and most GCD methods often suffer from the computational inefficiency. In spite of these limitations, local community detection(in short as LCD henceforth) would be very useful. Several LCD methods have been proposed to find the community containing a particular starting vertex for decades. According to the ways of how to evaluate the quality of a local community, the existing approaches to LCD can be classified into two main categories, namely, degree-based methods and similarity-based methods. 1)Degree-based methods [2,6,1,7,14] evaluate the local community quality by investigating vertices' degrees. 2)Similarity-based methods utilize similarities between vertices to help evaluate the local community quality [9,12,5]. Although LCD methods based degree and similarity are extensively studied, further study is still needed on finding a nice balance between the high efficiency of local search models and the high accuracy of detected communities. And how to ingeniously use the results of LCD to discovery the global non-overlapping and overlapping community structures is worth an in-depth study and concern.

In this work, we attempt to design a novel similarity-based method(LCE) for extracting local communities from large-scaled networks. In particular, LCE picks the neighbor vertex with the largest structure similarity as the candidate vertex and calculate the modularity gain to determine whether it should be added to the local community or not. Our method is naturally a heuristic, since

it does not examine all of vertices in the network, and the structural similarity of each pair of vertices in LCE is calculated only once by using a dynamical priority queue. So the execution of LCE is accelerated and the accuracy remains high. We further expand LCE for the global non-overlapping community detection, in which the labels of detected local communities are used as vertices' attributive tags. Finally, we adopt the results of LCE to calculate a membership matrix, which can been used to detect the global overlapping community of a graph. Experimental results on four real-life networks demonstrate the superiority of LCE over the classic degree-based and similarity-based LCD methods by either effectiveness or efficiency validity.

2 Problem Definition and Preliminaries

Let $\mathcal{G} = (V, E, w)$ be a given weighted undirected graph, where V is the set of nodes ($|V| = n$), E is the set of edges ($|E| = m$) that connect the nodes in V, and w is the weight of every edge in E. LCD is formulated as finding a subset of graph $\mathcal{G}' = (V', E')$ from a seed $v_s \in V$. Note that in LCD, the entire network structure is unknown at the beginning. Besides the detected local community, only partial information, i.e., the local community's neighbors and their linkage information, are available after each detection process. To be specific, we divide the explored graph into three regions: the local community $\mathcal{C}$, the boundary area $\mathcal{B}$ and a larger unknown area $\mathcal{U}$. Initially, we add the seed v_s to $\mathcal{C}$. Then, all of neighbors of nodes in $\mathcal{C}$ (e.g., v_s) are added to $\mathcal{B}$. In such local search model, $\mathcal{C}$ can be locally expanded from v_s with a predefined criterion.

Generally, a community is measured by a specific property of the vertices within it. For this task, different community measurements have been proposed [16,1,7,14] in recent years. In this paper, we adopt a structural similarity measure from the cosine similarity function [9] to effectively denotes the local connectivity density of any two adjacent vertices in a weighted network. Here, we first formalize some notions of the local community.

Definition 1 (Structural Similarity). *The structural similarity between two adjacent vertices v_i and v_j is defined as $s_{i,j} = \frac{\sum_{v_k \in \Gamma(v_i) \cap \Gamma(v_j)} w_{i,k} w_{j,k}}{\sum_{v_k \in \Gamma(v_i)} w_{i,k}^2 \sum_{v_k \in \Gamma(v_j)} w_{j,k}^2}$, where $\Gamma(v_i) = \{v_j \in V | \{v_i, v_j\} \in E\}$.*

The criterion we use to extract the local community containing the seed v_s is derived from [18], which finds a community with a large number of edges within itself and a small number of edges to the rest of the network.

Definition 2 (Local Modularity). *The local modularity of a community $\mathcal{C}$, denoted as $W(\mathcal{C})$, is given as $W(\mathcal{C}) = \frac{I(\mathcal{C})}{|\mathcal{C}|^2} - \frac{O(\mathcal{C})}{|\mathcal{C}||\mathcal{C}^c|}$, where $\mathcal{C}^c$ is the complement of $\mathcal{C}$, $I(\mathcal{C}) = \sum_{v_i, v_j \in \mathcal{C}} A_{ij}$, $O(\mathcal{C}) = \sum_{v_i \in \mathcal{C}, v_j \in \mathcal{C}^c} A_{ij}$, $A = [A_{ij}]$ is an $n \times n$ adjacency matrix of the graph $\mathcal{G}$.*

Based on the definition of local modularity, we have the following theorem.

Theorem 1. *The local modularity value of a community $\mathcal{C}$ will increase when $\mathcal{C}$ has high intra-cluster density and low inter-cluster density.*

Proof:The term $I(\mathcal{C})$ is twice the number of the edges within $\mathcal{C}$, and $O(\mathcal{C})$ represents the number of edges between $\mathcal{C}$ and the rest of the network. Each term is normalized by the total number of possible edges in each case. Note that we normalize the first term by $|\mathcal{C}|^2$ rather than $|\mathcal{C}|(|\mathcal{C}|-1)$ in order to conveniently derive the modularity gain discussed below, but in practice this makes little difference. Subject to this small difference, the local modularity can be described as the intra-cluster density minus the inter-cluster density. □

In Definition 2, we make an adjustment in the spirit of the ratio cut as $\hat{W}(\mathcal{C}) = |\mathcal{C}||\mathcal{C}^c|(\frac{I(\mathcal{C})}{|\mathcal{C}|^2} - \frac{O(\mathcal{C})}{|\mathcal{C}||\mathcal{C}^c|})$, where the factor $|\mathcal{C}||\mathcal{C}^c|$ penalizes very small and very large communities and produces more balanced solutions.

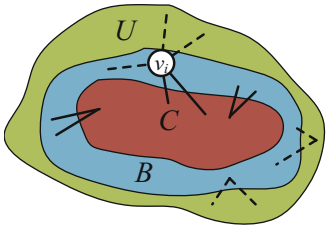


Fig. 1. The $\hat{W}$ variant when a vertex v_i joins $\mathcal{C}$

Suppose a community $\mathcal{C}$ is detected from a certain vertex v_s. We explore the adjacent vertices in the boundary area $\mathcal{B}$ of $\mathcal{C}$, as shown in Fig. 1. We distinguish three types of links: those internal to the community $\mathcal{C}(L)$, between $\mathcal{C}$ and the vertex $v_i(L_{in})$, between $\mathcal{C}$ and others vertices in $\mathcal{B}(L_{out})$. To simplify the calculations, we express the number of external links in terms of L and k_i(the degree of vertex v_i), so $L_{in} = a_1 L = a_2 k_i$, $L_{out} = b_1 L$, with $b_1 \geq 0$, $a_1 \geq \frac{1}{L}$, $a_2 \geq \frac{1}{k_i}$(since any v_i in $\mathcal{B}$ at least has one neighbor in $\mathcal{C}$). So, the value of $\hat{W}$ for the current community can be written as $\hat{W}(\mathcal{C}) = \frac{n-|\mathcal{C}|}{|\mathcal{C}|}2L - (a_1 + b_1)L$.

Definition 3 (Modularity Gain). *The modularity gain for the community $\mathcal{C}$ adopting a neighbor vertex v_i can be denoted as:*

$$\begin{aligned}\triangle\hat{W}_{\mathcal{C}}(v_i) &= (\frac{n-|\mathcal{C}|-1}{|\mathcal{C}|+1}2L(1+a_1) - (b_1 L + k_i - a_2 k_i)) - (\frac{n-|\mathcal{C}|}{|\mathcal{C}|}2L - (a_1+b_1)L) \\ &= 2n\frac{a_2 k_i|\mathcal{C}| - L}{|\mathcal{C}|(|\mathcal{C}|+1)} - k_i. \end{aligned} \quad (1)$$

$\triangle\hat{W}_{\mathcal{C}}(v_i)$ can be utilized as a criterion to determine whether the candidate vertex v_i should be included in the community $\mathcal{C}$ or not.

3 Local Community Extraction(LCE)

In this section, we propose a Local Community Extraction algorithm (in short as LCE henceforth). First, we introduce the basic idea of LCE and then present algorithmic details including the complexity analysis for LCE. Second, we introduce how to use LCE to detect the global non-overlapping and overlapping community structures.

To find the densely connected local community containing vertex v_s, LCE works with two iterative steps: update step and join step. First, the starting vertex v_s is added in $\mathcal{C}$. In the update step, LCE refreshes the the boundary area $\mathcal{B}$, and calculate the structural similarities between vertices in the community $\mathcal{C}$ and their neighbor vertices in $\mathcal{B}$. In the joining step, LCE tries to absorb a vertex in $\mathcal{B}$ having highest structural similarity with vertices in $\mathcal{C}$. If $\triangle\hat{W}_{\mathcal{C}}(v_i) > 0$, then the vertex v_i will be inserted into $\mathcal{C}$. Otherwise, it will be removed from $\mathcal{B}$. The two procedures above will be repeated in turn until set $\mathcal{B}$ is empty. Then, the whole community $\mathcal{C}(v_s)$ is discovered. We further select the vertex with maximum degree in $\mathcal{C}(v_s)$ as the core vertex, which can be also seen as the label of detected community. The pseudo-code of LCE is given in the following.

Algorithm 1. LCE(v_s)

Require: v_s
Ensure: $\mathcal{C}(v_s)$, $l(\mathcal{C}(v_s))$
1: $\mathcal{C} \leftarrow \{v_s\}$
2: $\mathcal{B} \leftarrow \{v_i | v_i \in \Gamma(v_s)\}$
3: **while** $\mathcal{B} \neq \emptyset$ **do**
4: $v_i^* = \arg\, max_{v_i \in \mathcal{B}} \sum_{v_j \in \mathcal{C}} s_{i,j}$
5: **if** $\triangle\hat{W}_{\mathcal{C}}(v_i^*) > 0$ **then**
6: $\mathcal{C} \leftarrow \mathcal{C} \cup \{v_i^*\}$
7: $\mathcal{B} \leftarrow \mathcal{B} \cup \{v_j' | v_j' \in \Gamma(v_i^*), v_j' \notin \mathcal{C}\}$
8: **else**
9: $\mathcal{B} \leftarrow \mathcal{B} - \{v_i^*\}$
10: **end if**
11: **end while**
12: $\mathcal{C}(v_s) \leftarrow \mathcal{C}$
13: $l(\mathcal{C}(v_s)) \leftarrow \arg\, max_{v_j \in \mathcal{C}} k_j$

Remark. Unlike existing methods [1,7,14], which calculate the quantitative metrics for each vertex in $\mathcal{B}$ and select the vertex who produces the greatest increment of the metric to join $\mathcal{B}$, LCE picks the neighbor vertex with the largest structure similarity as the candidate vertex v_i^* and calculate $\triangle\hat{W}_{\mathcal{C}}(v_i^*)$ to determine whether it should be added to $\mathcal{C}$ or not. The structural similarity reflects the local connectivity density of the graph. The larger the similarity between a vertex inside $\mathcal{C}$ and a vertex outside it, the more common neighbors the two vertices share, and the more probability they are at the same community. Furthermore, the structural similarity of each pair of vertices in LCE is calculated only once

by using a dynamical priority queue. So the execution of LCE is accelerated and the accuracy remains high.

Complexity Analysis. The running time of LCE is mainly consumed in line 4 of Algorithm 1. We can implement it using a binary Fibonacci heap H [9], which takes two steps: 1) Extract Step (extract the maximum element from H). As each Extract operation of H takes $O(logn')$ time and the body of the while loop is executed n' times, the total time for all Extract Steps is $O(n'logn')$, where n' is the number of vertices inferred(vertices in $\mathcal{C} \cup \mathcal{B}$). 2) Update Step (for each vertex in current $\mathcal{B}$, we update its sum of structure similarities with vertices in $\mathcal{C}$). First, the sum of structure similarities with vertices in $\mathcal{C}$ for each vertex $v_i \in \mathcal{B}$ should be computed, which can be completed in $O(k')$ time, where k' is the mean degree of inferred vertices. For vertices which are not in H, we insert them to H in $O(1)$ time; otherwise, it takes $O(1)$ time to make an Increase-Key operation. As the above steps are executed $O(m')$ times, where m' is the number of edges in $\mathcal{C} \cup \mathcal{B}$. Therefore, the total time of the Update Step is $O(m'k')$. Adding all together, the total time complexity is $O(m'k' + n'logn')$ for LCE.

Non-overlapping Community Detection. Non-overlapping community detection aims to find a good K-way partition $\mathcal{P} = \{\mathcal{C}_1, \cdots, \mathcal{C}_K\}$, where $\mathcal{C}_k$ is the k-th community, and $\mathcal{C}_1 \cup \cdots \cup \mathcal{C}_K \subseteq V$, $\mathcal{C}_k \bigcap \mathcal{C}_{k'} = \emptyset \ \forall \ k \neq k'$. Our assumption is that LCE inputted with similar adjacent vertices will return analogous community structures, in which the core vertices are almost unanimous. Therefore, if LCE returns the the same community label, the input vertices are likely to be in the same community. The process of LCE expansion algorithm for non-overlapping (in short as LCEnO henceforth) is given in Algorithm 2. Note that, the line 2 can be paralleled executed. Therefore, LCEnO could be completed in $O(m^*k^* + n^*logn^*)$ time, where n^*, m^* are the number of vertices and edges in the largest $\mathcal{C} \cup \mathcal{B}$, and k^* is the mean degree of inferred vertices in it.

Algorithm 2. LCEnO($\mathcal{G}$)

Require: $\mathcal{G} = (V, E, w)$
Ensure: $L = [l_s], s = 1, \cdots, n$
1: **for** $s = 1$; $s <= n$; $s++$ **do**
2: $[\mathcal{C}(v_s), l_s] \leftarrow$ LCE(v_s)// Parallel Computing
3: **end for**

Overlapping Community Detection. For an overlapping partition, overlapping communities can be represented as a membership matrix $\mathbf{U} = [u_{i,k}], i = 1, \cdots, n, k = 1, \cdots, K$, where $0 \leq u_{i,k} \leq 1$ denotes the ratio of membership that node i belongs to C_k. If node i belongs to only one community, $u_{i,k} = 1$, and it clearly follows that $\sum_{k=1}^{K} u_{i,k} = 1$ for all $1 \leq i \leq n$. With the detected communities of LCE, $u_{i,k}$ can be calculated as follows:

$$u_{i,k} = \frac{\sum_{s=1,\cdots,n \bigwedge l_s = k} \chi(v_i, \mathcal{C}(v_s))}{\sum_{s=1,\cdots,n} \chi(v_i, \mathcal{C}(v_s))}, \quad \chi(v_i, \mathcal{C}(v_s)) = \begin{cases} 1 & if v_i \in \mathcal{C}(v_s) \\ 0 & otherwise. \end{cases} \quad (2)$$

The process of LCE expansion algorithm for overlapping (in short as LCEO henceforth) is given as follows. The running time of LCEO is mainly consumed in lines 4-7 of Algorithm 3, which is calculating the membership that node i belongs to C_k. The total time of those steps is $O(nK)$. Adding the local community extraction steps, the total time complexity is $O(m^*k^*+n^*logn^*+nK)$ for LCEO.

Algorithm 3. LCEO($\mathcal{G}$)

Require: $\mathcal{G} = (V, E, w)$
Ensure: $\mathbf{U} = [u_{i,k}], i = 1, \cdots, n, k = 1, \cdots, K$
1: **for** $s = 1$; $s <= n$; $s++$ **do**
2: $[\mathcal{C}(v_s), l_s] \leftarrow$ LCE(v_s)// Parallel Computing
3: **end for**
4: **for** $i = 1$; $i <= n$; $i++$ **do**
5: **for** $k = 1$; $i <= K$; $k++$ **do**
6: $u_{i,k} \leftarrow \frac{\sum_{s=1,\cdots,n \bigwedge l_s = k} \chi(v_i, \mathcal{C}(v_s))}{\sum_{s=1,\cdots,n} \chi(v_i, \mathcal{C}(v_s))}$
7: **end for**
8: **end for**

We introduce the main framework of our approach by an example as shown in Fig. 2. The original graph includes 12 vertices and 20 edges. First, a state-of-the-art GCD algorithm known as FUC [3] is applied to identify its communities, the community structure is shown below the original graph in Fig. 2. Second, we employ LCE starting from all the vertices to detect their local communities. e.g., LCE starting from v_1 detects a local community including vertices v_1, v_5 and v_9, in which v_5 has the maximum degree. Therefore, the label of this community is marked as 5, which is also the attributive tag of vertex v_1. When we acquire all the attributive tags of 12 vertices, the global non-overlapping community structure of the graph has been detected. Finally, all local communities are assembled to be a membership matrix $\mathbf{U} = [u_{i,k}]$, which promulgates the global overlapping community structure.

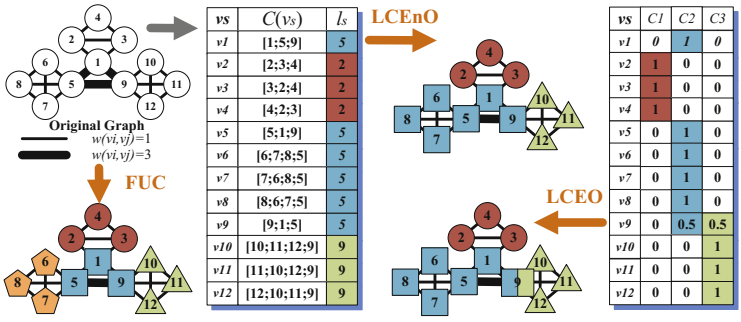


Fig. 2. An illustrative example

4 Experimental Results

Four real-world undirected networks: `Karate`, `NCAA`, `Facebook` and `PGP` are used for experiments. Some characteristics of these networks are shown in Table 1, where $|V|$ and $|E|$ indicate the numbers of nodes and edges respectively in the network, and $< k >$ indicates the average degree. `Karate` is a well known social network that describes the friendship relations between members of a karate club. `NCAA` is a representation of the schedule of American Division I college football games. Vertices in the network represent teams, which are divided into eleven communities(or conferences) and five independent teams. Edges represent regular season games between the two teams they connect. `Facebook` has been anonymized by replacing the Facebook-internal ids for each user with a new value. Each edge tells whether two users have the same political affiliations. `PGP` is a large scale social network, where each node represents a peer and each tie points out that one peer trusts the other.

Table 1. Real-world networks

Network	$\|V\|$	$\|E\|$	$< k >$
`Karate`	34	78	4.59
`NCAA`	115	616	10.71
`Facebook`	4,039	88,234	43.69
`PGP`	10,680	24,340	4.56

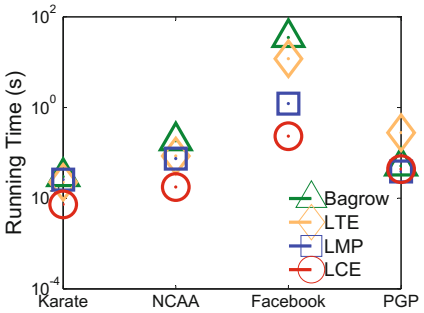


Fig. 3. Comparison on efficiency

4.1 Performance of LCE

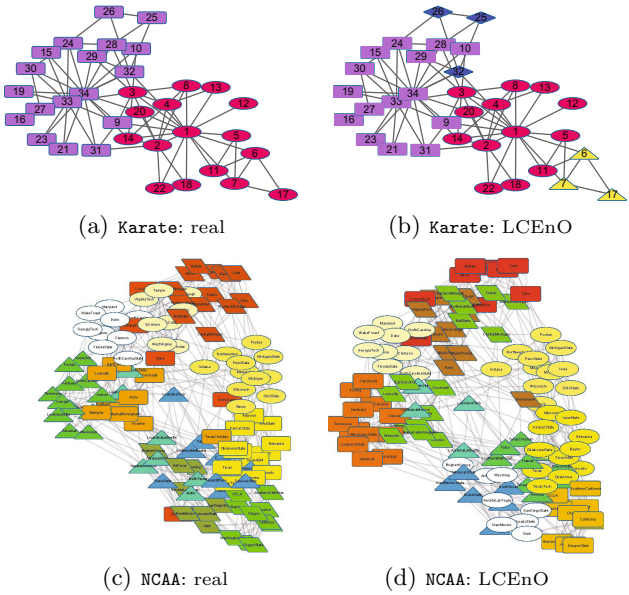
The Effectiveness. To test the effectiveness of LCE, the results are compared with the ground truth communities of each network. To be special, let $\mathcal{T}(v_s)$ be the ground truth community including the vertex v_i, we can compare $\mathcal{T}(v_s)$ and $\mathcal{C}(v_s)$ in the framework of Precision, Recall and F-measure (PRF) to assess our results. A higher value of precision (P) indicates fewer wrong classifications, while a higher value of recall (R) indicates less false negatives. It is common to use the harmonic mean of both measurements, called F-measure, which weighs precision and recall equally important. They are calculated as follows:

$$P(v_s)=\frac{|\mathcal{C}(v_s)\cap\mathcal{T}(v_s)|}{|\mathcal{C}(v_s)|},\ R(v_s)=\frac{|\mathcal{C}(v_s)\cap\mathcal{T}(v_s)|}{|\mathcal{T}(v_s)|},\ F1(v_s)=\frac{2P(v_s)R(v_s)}{P(v_s)+R(v_s)}. \quad (3)$$

Since the last two networks (`Facebook` and `PGP`) have no ground truth, we apply FUC [3] to identify communities of them, and utilize its detection results as the ground truth for the LCD algorithms. This is based on the intuition that a LCD method is acceptable if it can achieve an approximate result as a GCD approach does, because LCD methods usually perform faster than GCD

approaches. As the global community quality metrics such as the well-known Modularity metric [15] are not suitable to evaluate the quality of the detected local community, we use each vertex in a community as a seed and report algorithms' average precision, recall and F1-measure. We compare LCE with classical LCD algorithms, such as LWP [14], ELC [1], LTE [9]. The comparison results are presented in Table 2, from which we can observe that: 1)the recall values for all methods are overall worse than precision values, this is because LCD methods are based on the greedy search, which will trend to find a local optimal solution; 2)LCE almost achieves the high precision for all datasets, which demonstrate the superiority of its local search model over the other methods; 3)LCE usually outperforms LMR and ELC, and have a slight advantage over LTE, even though the later has been proven by extensive experiments to be one of the most accurate algorithms among previous LCD methods in[9].

The Efficiency. Fig. 3 shows the average running time of LDC methods starting from each vertex in the four test graphs. Apparently, the execution of LCE is more accelerated. Both LCE and LTE are similarity-based algorithms, their difference lies at the definition of local modularity. Compared with LCE, the calculation of modularity gain in LTE is more complex, which will consume extra time. LMR and ELC are degree-based LCD algorithms, which need calculate the quantitative metrics for each vertex in $\mathcal{B}$. The metric calculations are somewhat duplicate, which can not be simplified. Especially, the stopping criteria for ELC is to jude whether the current community is a "p-strong community", which will cost more time in every search step.



(a) `Karate`: real (b) `Karate`: LCEnO

(c) `NCAA`: real (d) `NCAA`: LCEnO

Fig. 4. LCEnO on small social networks

Table 2. Accuracy comparison of LCD on real-world networks

Network Comm.		LCE			LWP			ELC			LTE		
	size	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Karate-A	16	1.00	0.58	**0.73**	0.94	0.49	0.64	0.93	0.49	0.64	1.00	0.49	0.66
Karate-B	18	0.97	0.47	0.63	0.97	0.44	0.61	0.89	0.48	0.63	1.00	0.57	**0.73**
NCAA-AC	9	1.00	1.00	**1.00**	0.70	0.48	0.57	0.68	0.56	0.61	1.00	1.00	**1.00**
NCAA-BE	8	1.00	1.00	**1.00**	0.48	0.47	0.48	0.51	0.67	0.58	0.80	1.00	0.89
NCAA-Ten	12	1.00	1.00	**1.00**	0.33	0.26	0.29	0.17	0.21	0.19	1.00	1.00	**1.00**
NCAA-SE	12	1.00	1.00	**1.00**	0.81	0.55	0.65	0.83	0.85	0.84	1.00	1.00	**1.00**
NCAA-PT	10	0.91	0.82	**0.86**	0.68	0.58	0.62	0.68	0.73	0.70	0.91	0.82	**0.86**
NCAA-Others	5	0.12	0.24	0.16	0.21	0.40	**0.27**	0.14	0.52	0.22	0.19	0.32	0.24
NCAA-MA	13	1.00	0.50	0.67	0.78	0.48	0.60	0.81	0.78	**0.79**	0.86	0.50	0.64
NCAA-MV	8	1.00	1.00	**1.00**	0.76	0.70	0.73	0.67	0.70	0.69	1.00	1.00	**1.00**
NCAA-WA	10	1.00	1.00	**1.00**	0.65	0.45	0.53	0.67	0.60	0.63	1.00	1.00	**1.00**
NCAA-Twelve	12	1.00	1.00	**1.00**	0.67	0.40	0.52	0.61	0.56	0.35	1.00	1.00	**1.00**
NCAA-SB	7	0.64	0.51	**0.57**	0.49	0.61	0.54	0.23	0.69	0.35	0.64	0.51	0.56
NCAA-USA	10	0.74	0.66	**0.70**	0.41	0.32	0.36	0.25	0.23	0.24	0.74	0.66	**0.70**
Facebook-1	341	1.00	0.16	0.28	0.99	0.05	0.10	0.88	**0.40**	0.55	1.00	0.15	0.26
Facebook-2	66	0.88	0.48	0.61	0.42	0.14	0.21	0.16	0.96	0.27	0.94	0.57	**0.71**
Facebook-3	308	0.94	0.26	0.41	0.92	0.07	0.13	0.41	0.15	0.22	0.97	0.18	**0.30**
Facebook-4	25	0.96	1.00	0.98	1.00	0.36	0.53	0.97	0.59	0.74	1.00	1.00	**1.00**
Facebook-5	206	1.00	0.33	**0.50**	1.00	0.09	0.17	0.97	0.31	0.47	1.00	0.33	0.49
Facebook-6	62	0.94	0.42	0.58	0.90	0.19	0.31	0.56	0.32	0.41	0.99	0.44	**0.61**
Facebook-7	408	0.94	0.58	0.71	0.38	0.04	0.07	0.21	0.17	0.18	0.96	0.60	**0.74**
Facebook-8	483	0.94	0.19	**0.31**	0.81	0.05	0.09	0.16	0.13	0.14	0.97	0.16	0.27
Facebook-9	442	0.98	0.30	**0.45**	0.97	0.07	0.14	0.97	0.20	0.33	1.00	0.24	0.38
Facebook-10	73	0.94	0.92	0.93	0.53	0.19	0.28	0.06	0.12	0.08	1.00	1.00	**1.00**
Facebook-11	237	0.99	0.87	**0.92**	0.26	0.07	0.10	0.15	0.03	0.04	1.00	0.82	0.90
Facebook-12	226	0.98	0.68	**0.80**	0.96	0.13	0.23	0.10	0.21	0.14	0.99	0.46	0.63
Facebook-13	554	0.98	0.18	0.31	0.96	0.06	0.10	0.63	0.37	**0.46**	0.99	0.18	0.21
Facebook-14	548	1.00	0.11	0.20	0.99	0.03	0.07	0.98	0.24	**0.39**	1.00	0.08	0.12
Facebook-15	60	1.00	0.33	**0.50**	0.98	0.13	0.23	0.99	0.33	**0.50**	0.98	0.14	0.24
PGP-1	395	0.95	0.16	**0.28**	0.86	0.16	0.27	0.82	0.12	0.21	0.96	0.12	0.21
PGP-2	303	0.93	0.22	**0.36**	0.92	0.22	**0.36**	0.73	0.19	0.30	0.93	0.19	0.32
PGP-3	974	0.94	0.13	0.24	0.74	0.13	0.22	0.84	0.17	**0.29**	0.94	0.18	0.31
PGP-4	379	0.99	0.12	0.21	0.78	0.12	0.20	0.90	0.17	**0.29**	0.99	0.13	0.21
PGP-5	1457	0.93	0.11	**0.20**	0.88	0.11	**0.20**	0.76	0.08	0.15	0.93	0.09	0.17
PGP-6	798	0.98	0.06	0.11	0.94	0.06	0.11	1.00	0.08	**0.16**	0.98	0.08	0.15
PGP-7	1289	0.96	0.14	0.24	0.80	0.14	0.23	0.76	0.13	0.22	0.96	0.17	**0.29**
PGP-8	513	0.97	0.17	**0.29**	0.87	0.17	0.28	0.87	0.11	0.20	0.97	0.11	0.20
PGP-9	417	0.93	0.17	0.28	0.93	0.17	0.28	0.92	0.27	**0.41**	0.92	0.13	0.22
PGP-10	1091	0.93	0.22	0.36	0.86	0.22	0.35	0.83	0.19	0.31	0.93	0.31	**0.47**

4.2 Performance of LCEnO

Here, we first apply LCEnO to the two small social networks with ground truth: Karate and NCAA. The purpose is to gain a direct understanding of non-

overlapping community detection by network visualization. Then, we further compare LCEnO with classical GCD methods, such as FNM [15], FUC [3], Metis [10], and Cluto [11].

`Karate` is split into two parties following a disagreement between an instructor (node 1) and an administrator (node 34), which serves as the ground truth about the communities in Fig. 4(a). We employ LCEnO to extract non-overlapping communities from the network. The result is shown in Fig. 4(b), which supplements the division of the club with more information. More interestingly, LCEnO actually tends to partition this network into four rather than two communities, as indicated by the nodes in four colors/shapes in Fig. 4(b). This implies that there exits a latent sub-party (including vertices 6, 7, 11) inside the party led by node 1, and a latent sub-party (including vertices 25, 26, 32) inside the party led by node 34.

The ground truth of `NCAA` labels vertices with their actual conferences, corresponding twelve different colors/shapes in Fig. 4(c). As shown in Fig. 4(d), LCEnO generally well captures the "sharp-cut" teams in conferences "AC", "BE", "Ten", "SE", "MV", "WA", and "Twelve" respectively, although there yet exists some teams assigned mistakenly. Note that nearly all the "Orangered rectangle" in Fig. 4(c) are totally detected mistakenly by LCEnO. This is indeed reasonable since those vertices have very few internal connections, actually, they represent five independent teams (Utah State, Navy, Notre Dame, Connecticut and Central Florida) in NCAA.

We compare LCEnO with GCD methods, such as FNM [15], FUC [3], Metis [10], and Cluto [11] on the effectiveness. For each method/network, Table 3 displays the modularity that is achieved and the running time. The modularity obtained by LCEnO are slightly lower than FUC's, but it outperforms nearly all the other methods. In terms of running time, Metis has a great advantage due to its parallel processing modules. However, it perform poor on graphs with obscure community structure, e.g., `Karate` and `NCAA`. While LCEnO keeps a nice balance between high modularity and short running time, which can be applied large scale network community detection.

Table 3. Modularity and running time comparison

Network	LCEnO	FNM	FUC	Metis	Cluto
`Karate`	0.38/0.03s	0.38/0.05s	**0.42**/0.03s	0.24/**0.01**s	0.36/0.02s
`NCAA`	0.58/0.20s	0.57/0.20s	**0.60**/0.06s	0.08/**0.01**s	0.60/0.03s
`Facebook`	0.73/2.68s	0.78/8.45m	**0.84**/6.29s	0.79/**0.53s**	0.82/4.24s
`PGP`	0.67/**0.44**s	0.85/179.42m	**0.88**/22.50s	0.83/1.76s	0.72/11.90s

4.3 Performance of LCEO

To evaluate the performance of LCEO, we also employ the PRF framework. Let $\hat{\mathcal{C}}_k$ be the k-th overlapping community, which obeys $\hat{\mathcal{C}}_1 \cup \cdots \cup \hat{\mathcal{C}}_K \subseteq V$. In the following, we introduce a membership threshold α, $0 < \alpha \leq 1$, to control the scale at which we want to observe the overlapping communities in a network.

Definition 4 (α-Overlapping Community). *The k-th α-overlapping community, denoted by $\hat{\mathcal{C}}_k(\alpha)$, is defined as $\hat{\mathcal{C}}_k(\alpha) = \{v_i | u_{i,k} \geq \alpha\}$.*

We can use each vertex in a overlapping community as a seed and report LCEO's average PRF. Fig. 5 shows the accuracy in the function of α for the four test graphs, from which we can observe that: 1)the recall values for LECHO have a significant improvement in all scales, compared with previous LCD algorithms; 2)the values of α in the range $[0.6, 0.8]$ are optimal, in the sense that overlapping communities extracted by LCEO in this region have a high F1-measure; 3)LECHO performs better in dense networks rather than in sparse networks.

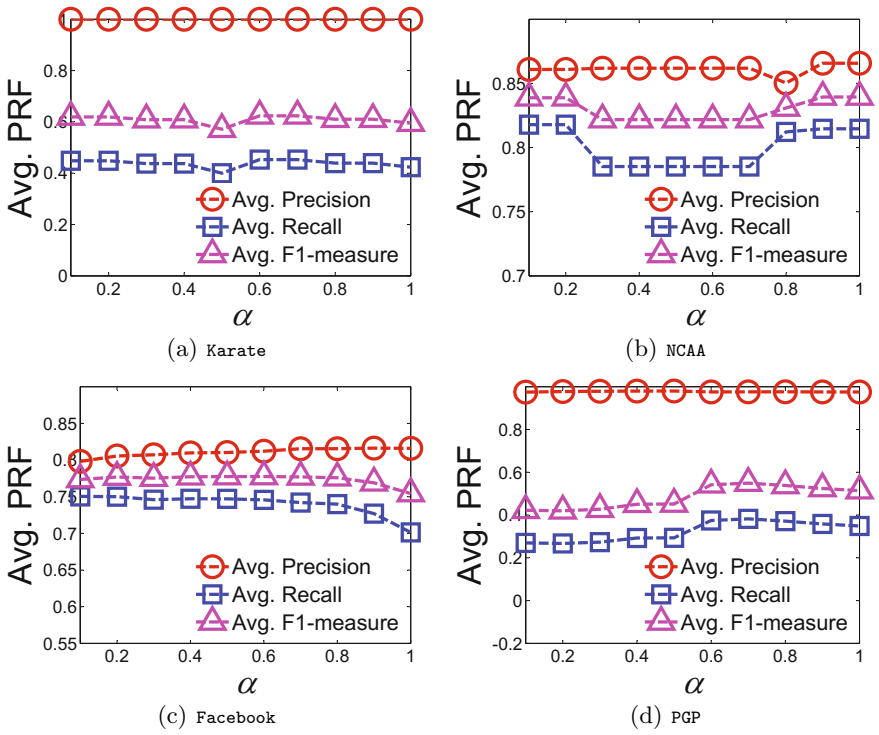


Fig. 5. The accuracy for different α on the four test networks

5 Related Work

Existing approaches to LCD can be classified into two main categories, namely, degree-based methods and similarity-based methods. 1)Degree-based methods evaluate the local community quality by investigating verticesdegrees. Some naive solutions, such as l-shell search algorithm [2], discovery-then-examination approach [6], and outwardness-based method [1], only consider the number of edges inside and outside a local community. Clauset [7] defines local modularity by considering the boundary points of a sub-graph, and proposes a greedy

algorithm on optimizing this measure. Similarly, Luo et al. [14] present another measurement as the ratio of the internal degree and external degree of a subgraph. Both methods can achieve high recall but suffer from low precision due to including many outliers [6]. 2)Similarity-based methods utilize similarities between vertices to help evaluate the local community quality. LTE algorithm [9] is a representative of this kind, using a well-designed metric for local community quality known as Tightness. There are a few alternative similarity-based metrics such as VSP [12] and RSS [5] that can also help evaluate the local community quality, although they are not originally designed for LCD.

Although LCD methods based degree and similarity are extensively studied, further study is still needed on finding a nice balance between the high efficiency of local search models and the high accuracy of detected communities. And how to ingeniously use the results of LCD to discovery the global non-overlapping and overlapping community structures is worth an in-depth study and concern. Our work attempts to fill this void by conducting community extraction based on an efficient LCE method.

6 Conclusion

This work proposes a Local Community Extraction algorithm (LCE) to find the local clusters from a seed vertex. First, a local search model is carefully designed to determine candidate vertices to be preserved or discarded, which only relies on the local knowledge rather than the global view of the network. Second, we expand LCE for the global non-overlapping community detection, in which we use the labels of detected local communities as vertices' attributive tags. Finally, we use the results of LCE to calculate the membership matrix, which can been guided for the global overlapping community detection. Experimental results on real-life networks demonstrate the advantage of LCE over the classic degree-based and similarity-based LCD methods by either effectiveness or efficiency.

Acknowledgment. This research was partially supported by NSFC (Nos. 71372188, 61103229), National Center for International Joint Research on E-Business Information Processing (No. 2013B01035), National Key Technologies R&D Program of China (No. 2013BAH16F01), Industry Projects in Jiangsu S&T Pillar Program (No. BE2012185), the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD),and Key/Surface Project of Natural Science Research in Jiangsu Provincial Colleges and Universities (Nos. 12KJA520001, 14KJA520001, 14KJB520015).

References

1. Bagrow, J.P.: Evaluating local community methods in networks. Journal of Statistical Mechanics: Theory and Experiment 2008(05), 5001 (2008)
2. Bagrow, J.P., Bollt, E.M.: Local method for detecting communities. Physical Review E 72(4), 46108 (2005)

3. Blondel, V.D., Guillaume, J.-L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment 2008(10), 10008 (2008)
4. Bu, Z., Zhang, C., Xia, Z., Wang, J.: A fast parallel modularity optimization algorithm (fpmqa) for community detection in online social network. Knowledge-Based Systems 50, 246–259 (2013)
5. Chen, H.-H., Gou, L., Zhang, X.L., Giles, C.L.: Discovering missing links in networks using vertex similarity measures. In: Proceedings of the 27th Annual ACM Symposium on Applied Computing, pp. 138–143. ACM (2012)
6. Chen, J., Zaïane, O., Goebel, R.: Local community identification in social networks. In: International Conference on Advances Social Network Analysis and Mining, ASONAM 2009, pp. 237–242. IEEE (2009)
7. Clauset, A.: Finding local community structure in networks. Physical Review E 72(2), 26132 (2005)
8. Hlaoui, A., Wang, S.: A direct approach to graph clustering. In: Neural Networks and Computational Intelligence, pp. 158–163 (2004)
9. Huang, J., Sun, H., Liu, Y., Song, Q., Weninger, T.: Towards online multiresolution community detection in large-scale networks. PloS one 6(8), e23829 (2011)
10. Karypis, G.: Multi-constraint mesh partitioning for contact/impact computations. In: Proceedings of the 2003 ACM/IEEE conference on Supercomputing, p. 56. ACM (2003)
11. Karypis, G., Han, E.H., Kumar, V.: Chameleon: Hierarchical clustering using dynamic modeling. Computer 32(8), 68–75 (1999)
12. Li, K., Pang, Y.: A vertex similarity probability model for finding network community structure. In: Tan, P.-N., Chawla, S., Ho, C.K., Bailey, J. (eds.) PAKDD 2012, Part I. LNCS, vol. 7301, pp. 456–467. Springer, Heidelberg (2012)
13. Luccio, F., Sami, M.: On the decomposition of networks in minimally interconnected subnetworks. IEEE Transactions on Circuit Theory 16(2), 184–188 (1969)
14. Luo, F., Wang, J.Z., Promislow, E.: Exploring local community structures in large networks. Web Intelligence and Agent Systems 6(4), 387–400 (2008)
15. Newman, M.E., Girvan, M.: Finding and evaluating community structure in networks. Physical Review E 69(2), 26113 (2004)
16. Radicchi, F., Castellano, C., Cecconi, F., Loreto, V., Parisi, D.: Defining and identifying communities in networks. Proceedings of the National Academy of Sciences of the United States of America 101(9), 2658–2663 (2004)
17. Von Luxburg, U.: A tutorial on spectral clustering. Statistics and Computing 17(4), 395–416 (2007)
18. Zhao, Y., Levina, E., Zhu, J.: Community extraction for social networks. Proceedings of the National Academy of Sciences 108(18), 7321–7326 (2011)