Physica A 392 (2013) 3095–3105

Contents lists available at SciVerse ScienceDirect

Physica A

journal homepage: www.elsevier.com/locate/physa

Detecting community structure using label propagation with weighted coherent neighborhood propinquity

Hao Lou [1], Shenghong Li *, Yuxin Zhao

School of Electronic Information and Electrical Engineering, Shanghai Jiaotong University, Shanghai 200240, China

HIGHLIGHTS

- Label propagation algorithm is near linear time and effective in community detection.
- We extend the horizon of the node in label propagation algorithm to two or more hops.
- Propinquity measures the probability that two nodes are involved in a same community.
- Entropy based weight modulation is proposed for coherent neighborhood propinquity.
- The performance and stability in community detection are highly improved.

ARTICLE INFO

Article history:
Received 10 October 2012
Received in revised form 5 March 2013
Available online 29 March 2013

Keywords:
Community structure
Community detection
Label propagation
Coherent neighborhood propinquity

ABSTRACT

Community detection has become an important methodology to understand the organization and function of various real-world networks. The label propagation algorithm (LPA) is an almost linear time algorithm proved to be effective in finding a good community structure. However, LPA has a limitation caused by its one-hop horizon. Specifically, each node in LPA adopts the label shared by most of its one-hop neighbors; much network topology information is lost in this process, which we believe is one of the main reasons for its instability and poor performance. Therefore in this paper we introduce a measure named weighted coherent neighborhood propinquity (weighted-CNP) to represent the probability that a pair of vertices are involved in the same community. In label update, a node adopts the label that has the maximum weighted-CNP instead of the one that is shared by most of its neighbors. We propose a dynamic and adaptive weighted-CNP called entropic-CNP by using the principal of entropy to modulate the weights. Furthermore, we propose a framework to integrate the weighted-CNP in other algorithms in detecting community structure. We test our algorithm on both computer-generated networks and real-world networks. The experimental results show that our algorithm is more robust and effective than LPA in large-scale networks.

1. Introduction

Many complex systems, such as social communication networks, biological interaction networks and the Internet, can be modeled as networks, with vertices for entities and edges for relations between them. One of the most important properties for complex networks is community structure which is usually a group of nodes that are densely inter-connected and sparsely connected to other parts [1–3]. It has much practical significance when such communities represent organized groups or functional modules with nodes of common features. For instance, groups within the world-wide web might

* Corresponding author.
E-mail addresses: louhaozixin@gmail.com (H. Lou), shli@sjtu.edu.cn (S. Li).
[1] Tel.: +86 13917416326.

0378-4371/$ – see front matter
http://dx.doi.org/10.1016/j.physa.2013.03.014

correspond to sets of web pages on the same topic [4]. Therefore, community detection provides an important insight into the topology, organization and functional behavior of the real-world complex systems.

Algorithms for community detection, also called community clustering, have garnered significant attention in the past decade. In the literature, several different approaches have been proposed to find community structure in networks. Traditional graph partitioning methods such as the Kernighan–Lin algorithm [5] and spectral bisection method [6] had been extended for community detection. The Kernighan–Lin algorithm attempted to minimize the difference between intra-connected edges and inter-connected edges to detect communities. The spectral bisection method exploiting the spectral property of the Laplacian matrix [7] or normal matrix [8] had been used to divide a network into two groups so that the number of edges between groups is minimized, i.e. "minimum cut". Hierarchical clustering algorithms are another popular class of methods. Hierarchical clustering techniques aim at identifying groups of vertices with high similarity, and can be classified in two categories, i.e. agglomerative algorithms and divisive algorithms. The most popular algorithm proposed by Girvan and Newman [2] is a hierarchical divisive algorithm, in which edges are iteratively removed based on the value of their betweenness, which expresses the number of shortest paths between pairs of nodes that pass through the edge. Newman and Girvan first introduced the quality function *Modularity* Q to define a stop criterion for the algorithm in Ref. [9] to detect community. And since then *Modularity* has rapidly become an essential element of many clustering methods. Algorithms like greedy optimization in Ref. [9], simulated annealing in Ref. [10], extremal optimization in Ref. [11] and spectral optimization in Ref. [12], try to optimize the modularity function to detect community structure. CPM algorithm [13] can identify overlapping communities based on the assumption that a community is composed of numbers of adjacent k-cliques. The structural algorithm [14] and Dynamic algorithm [15] by Rosvall and Bergstrom turned the problem into optimally compressing the information on the structure of the graph, so that one can recover as closely as possible the original structure when the compressed information is decoded. They are indeed effective in finding a good community structure. The Potts model approach [16] by Ronhovde and Nussinov is based on the minimization of the Hamiltonian of a Potts-like spin model, where the spin state represents the membership of the node in a given community. The method is rather fast and its complexity is slightly super-linear.

However, most algorithms mentioned above are limited when used in very large scale networks because of high time complexity or no *priori* knowledge. For example, the GN algorithm [2] requires that a centrality score is computed for each edge. This centralized and global control is a significant bottleneck. The label propagation algorithm (LPA) [17] proposed by Raghavan et al. shows good prospects. It employs the diffusion of information to identify communities. Every node is initialized with a unique label and at every step each node adopts the label that most of its neighbors currently have. In this iterative process densely connected groups of nodes form a consensus on a unique label to form community. This agent-based and decentralized approach brings almost linear time complexity, requires no priori information and is suitable for large-scale networks with millions of nodes and edges [18].

Nevertheless, LPA adopts the label that is shared by most of its neighbors, which means the horizon of a node is only one hop and the further connections of its neighbors are ignored, causing the loss of much topological information of the network. Realizing this shortage, in this paper we introduce the coherent neighborhood propinquity (CNP) defined in Ref. [19] and extend it to weighted-CNP. Integrating this weighted-CNP into LPA, we propose a community detection algorithm named LPA-CNP whereby the label of a node is updated to the label that has the maximum weighted-CNP. We also propose an adaptive weighted-CNP called entropic-CNP which uses the principal of entropy to modulate the weights. We test our algorithm on both computer-generated benchmark networks and real-world networks. The results show that our LPA-CNP performs better than LPA especially in large-scale networks and robustness is significantly improved.

The remainder of the paper is organized as follows. In Section 2, we introduce the label propagation algorithm and then in Section 3 describe our LPA-CNP algorithm. The experimental results are presented in Sections 4 and 5 gives our conclusion.

2. Label propagation algorithm

In this paper, we focus our attention on unweighted, undirected networks. A complex network is represented by $G(V, E)$, where V is the vertex set and E is the edge set. Each node $v (v \in V)$ has a label c_v and $N(v)$ is the set of neighbors of node v. The label propagation algorithm (LPA) first initializes every node with a unique label. Then at every step each node updates its current label to the label shared by the maximum number of its neighbors, i.e.

$$c_v = \arg\max_l |N^l(v)| \tag{1}$$

where $N^l(v)$ is the set of neighbors of node v that have the label l, and $|X|$ is the cardinality of set X. By this iterative process densely connected groups of nodes form consensus on one label to form communities. Finally, LPA converges when no nodes change anymore. Nodes sharing the same label are classified into the same community. When there are multiple maximal labels among the neighbors' label, we adopt the solution of Raghavan et al. [17], i.e. if one of them equals the current node's label c_v, the node retains its original label. Otherwise, ties are broken randomly.

The biggest advantage of LPA is its near linear time complexity $O(km)$, where k is the number of iterations and m is the number of edges. Raghavan et al. [17] mentioned that 95% of nodes or more are classified correctly by the end of iteration

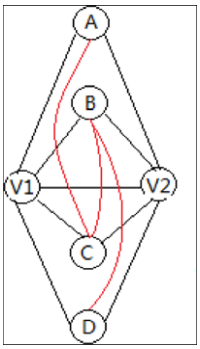

Fig. 1. Coherent neighbor propinquity.

5. Due to this simple local and decentralized process, LPA is applicable to large-scale networks with millions of nodes and edges.

Still, in order to solve the label oscillations to ensure convergence, Raghavan et al. [17] have proposed the asynchronous update of nodes in random order that severely hampers its robustness and the stability of the identified community structure [20]. The stability of LPA has become a severe issue and further improvement has been done in various ways [18,20–22]. Subelj et al. [20] proposed a *balanced propagation* that counteracts for the introduced randomness by utilizing *node balancers*. Leung et al. [21] proposed a *label hop attenuation* technique to prevent the label from spreading too far from its origin. Both of them result in a simple incorporation of *propagation preference* [21] into the update rule as

$$c_v = \arg\max_l \sum_{s \in N^l(v)} r_s \tag{2}$$

where r_s is the *propagation preference* of node s.

The above improvements indeed result in good performance and strong robustness and they mainly focus on the update order or node preference (or something else). However, reviewing the LPA carefully, we can find that the updating strategy loses much local topological information. On the one hand, LPA treats all the node's neighbors equally, neglecting the closeness of nodes. On the other hand, the horizon of the node in LPA is only one hop, causing the loss of topological information of further hops, which we think is another intrinsic reason for LPA's instability and poor performance. This issue has been ignored (or been neglected) in the past, therefore in this paper we focus on this point and propose a new algorithm to solve this issue and improve the performance of LPA.

3. Label propagation with weighted coherent neighborhood propinquity

3.1. Coherent neighborhood propinquity

Propinquity is a term used by sociologists. It refers to the physical or psychological proximity between people. Zhang et al. [19] used it to describe a quantity to evaluate the probability that a pair of vertices are involved in the same community. The network horizon is generally two or three hops [23]. In [24] the authors proved that the diameter of a coherent graph is no greater than 2. A coherent graph is a γ-quasi-clique where $\gamma = 0.5$ [25]. In a quasi-clique, each vertex is connected with at least a minimum proportion γ of other vertices. So Zhang et al. [19] suggest that the propinquity definition should consider only the local neighborhood within two hops and the propinquity between a pair of vertices with distance greater than 2 should be zero, if the resulting community structures are coherent (i.e. γ-quasi-clique where $\gamma \geq 0.5$). They also define the term *coherent neighborhood propinquity (CNP)* as:

$$P(v1, v2) = |E(v1, v2)| + |N(v1) \cap N(v2)| + |E(G[N(v1) \cap N(v2)])| \tag{3}$$

where $|E(v1, v2)|$ denotes the number of edges connecting node $v1$ and node $v2$. Here it is 0 if node $v1$ and node $v2$ have no edge and 1 otherwise. We call this item *direct propinquity*. The second item is the count of common neighbor vertices and named *angle propinquity*. The third part which is called *conjugate propinquity* is the number of edges connecting common neighbor vertices of $v1$ and $v2$. As Fig. 1 shows, the four nodes A, B, C, D are common neighbors contributing 4 to the CNP (*angle propinquity*) between node $v1$ and $v2$. The three edges in red which contribute 3 are *conjugate propinquity*. Adding up with the *direct propinquity*, the total CNP between $v1$ and $v2$ is 8.

This definition restricts the propinquity calculation in a controllable local scope which has acceptable cost in time complexity, while involving adequate local topological information of the network to reflect the probability that a pair of vertices are involved in the same community.

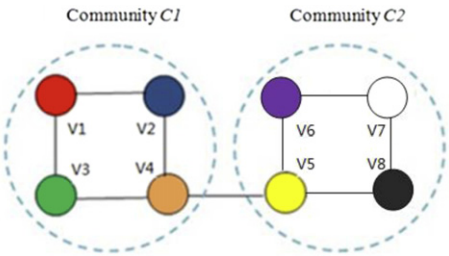

Fig. 2. Toy example network with two strong communities. Node colors indicate their community labels. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

3.2. Variant of CNP

As discussed above, CNP represents the similarity of two vertices. It involves three parts, *direct propinquity*, *angle propinquity* and *conjugate propinquity*. These three parts give many clues in CNP measurement. Each part can be viewed as a feature corresponding to one aspect of the measurement. CNP treats each feature equivalently. However, for different networks, each feature should have a different importance in CNP calculation. So weighted-CNP is more effective and reasonable to measure the similarity of two vertices in a certain network. We use the variant below to calculate the weighted-CNP.

$$P_w(v1, v2) = k_1 \times |E(v1, v2)| + k_2 \times |N(v1) \cap N(v2)| + k_3 \times |E(G[N(v1) \cap N(v2)])| \tag{4}$$

where $P_w(v1, v2)$ is the weighted-CNP between $v1$ and $v2$. k_1, k_2, k_3 are three weights for adjusting the importance of each feature in calculation.

Eq. (4) can be transformed to (5), as follows.

$$\begin{cases} P_w(v1, v2) = |E(v1, v2)| + w_1 \times |N(v1) \cap N(v2)| + w_2 \times |E(G[N(v1) \cap N(v2)])| \\ w_1 = \dfrac{k_2}{k_1} \\ w_2 = \dfrac{k_3}{k_1} \end{cases} \tag{5}$$

In (5), w_1 and w_2 are the relative weights for *angle propinquity* and *conjugate propinquity*. If k_1, k_2, k_3 are equivalent, i.e. w_1 and w_2 are equal to 1, then weighted-CNP is just the original definition of CNP. However, this is often not the case. w_1 and w_2 should be dynamic and adaptive to a given network.

3.3. Label propagation with weighted-CNP

The basic label propagation algorithm updates the label of every node to the label shared by most of its neighbors asynchronously and randomly. We study the label propagation algorithm on a toy sample network in Fig. 2. The network consists of two communities, namely C_1 and C_2, that accord with the definition of strong community (i.e., each node has more intra-community than inter-community edges). The colors of the nodes represent their labels.

If we first update node v_4, it will randomly choose one label from node v_2, node v_3 and node v_5. If unfortunately the label of node v_5 is selected, node v_4 becomes yellow. The label of Community C_2 (yellow) invades in Community C_1. Gradually nodes in Community C_1 and Community C_2 may all update to yellow, forming a trivial community. If node v_4 extends its horizon to two hops, the label of node v_5 will certainly not be considered when updating node v_4 because of the existence of node v_1. The label of Community C_2 can hardly invade into Community C_1. Therefore extending the horizon of a node in LPA is useful.

Weighted-CNP discussed above reflects the probability that a pair of nodes is involved in a same community, involving adequate local topological information, i.e. direct edge, common neighbors and the connection of common neighbors. It can exactly satisfy our need. We integrate weighted-CNP to LPA and call this improved algorithm LPA-CNP.

In LPA-CNP, at every step each node v updates its original label to the label which has the maximum weighted-CNP, i.e.

$$c_v = \arg\max_l \sum_{s \in V^l} P_w(v, s) \tag{6}$$

where $P_w(v, s)$ is the weighted-CNP of node v and node s. V^l is the set of nodes with label l in entire vertex set V. The nodes considered are not limited in the neighbor of node v. LPA-CNP can be described in the following steps.

(1) Initially assign each node in the network a unique label.
(2) Calculate the weighted-CNP for every pair of nodes by Eq. (5).
(3) Arrange the nodes in the network in a random order.

Table 1
All the CNPs of the toy example.

	V1	V2	V3	V4	V5	V6	V7	V8
V1	–	1	1	2	0	0	0	0
V2	1	–	2	1	1	0	0	0
V3	1	2	–	1	1	0	0	0
V4	2	1	1	–	1	1	0	1
V5	0	1	1	1	–	1	2	1
V6	0	0	0	1	1	–	1	2
V7	0	0	0	0	2	1	–	1
V8	0	0	0	1	1	2	1	–

(4) For each node chosen from the random order, update its label to the label which has the maximum weighted-CNP by Eq. (6) and ties are broken randomly.
(5) If every node has a label which has maximum weighted-CNP, then stop the algorithm. Else, go back to (3) and iterate the procedure.

Since weighted-CNP restricts the propinquity calculation in a controllable local scope of two hops, and the weighted-CNP of every pair of two nodes should be calculated only once, the time complexity is acceptable.

See the toy example again, and we use the CNP (i.e. w_1 and w_2 are equal to 1 in weighted-CNP). We can get all the CNPs by Eq. (3). The results are shown in Table 1. Again we first update node v_4, its label will update to the label of node v_1. The label of node v_5 cannot propagate to community C_1 easily, unless node v_6 or node v_8 has the same label with node v_5. The network is more likely to be divided into two communities correctly.

In weighted-CNP, if k_2 and k_3 are 0, then w_1 and w_2 become 0. The weighted-CNP between node v and another node which has no direct edge with node v is 0, otherwise is 1. And this LPA-CNP is just original LPA.

3.4. Entropic-CNP

As mentioned before, the weighted-CNP should be adaptive. The values of w_1 and w_2 change with different networks. One approach to obtain the values from the network is entropy based weight modulation [26].

We view the three parts of weighted-CNP as three features corresponding to three aspects of the measurement. Then the Feature-Pair matrix $M_{F\text{-}P}$ [26] is generated from calculating the values of each feature in different pairs. The row represents a feature and there are three rows in $M_{F\text{-}P}$ corresponding to the three parts *direct propinquity, angle propinquity*,and *conjugate propinquity*. The columns represent the node pairs. The element fp_{ij} in $M_{F\text{-}P}$ denotes the value of feature i of pair j. The distribution of a feature implies the importance of a feature. The more uniformly a feature distributes the less information it carries, thus less important. Hence, entropy can be borrowed to represent the information strength of a feature, and we call it *feature entropy* [26]. By definition, the *feature entropy* **FE** is:

$$FE(i) = -\sum_{j=1}^{n} p_{ij} \log_n p_{ij} \tag{7}$$

where i is the ith feature, j is the jth pair of nodes and p_{ij} is a normalized feature-frequency value. p_{ij} is obtained by

$$p_{ij} = fp_{ij} \Big/ \left(\sum_{k=1}^{n} fp_{ik} \right). \tag{8}$$

In (7) and (8), n is the cardinal of the set of pairs between each node in the network. The *feature entropy* **FE** denotes the relative significance of each part in weighted-CNP.

The selection of weight w_1 and w_2 depends upon the *inverse feature entropy* [26], **IFE**, which is inversely proportional to the *feature entropy* **FE**.

$$IFE(i) = \phi(FE(i)) \tag{9}$$

where ϕ denotes a negative correlation function. One alternative function is:

$$IFE(i) = \frac{1}{FE(i)}. \tag{10}$$

If FE_1, FE_2, FE_3 respectively represent the *feature entropy* of *direct propinquity, angle propinquity* and *conjugate propinquity*, then weights k_1, k_2, k_3 in weighted-CNP have the following relations:

$$k_1 : k_2 : k_3 = \frac{1}{FE_1} : \frac{1}{FE_2} : \frac{1}{FE_3}. \tag{11}$$

And then

$$w_1 = \frac{FE_1}{FE_2}, \qquad w_2 = \frac{FE_1}{FE_3} \tag{12}$$

where w_1 and w_2 are the relative weights in Eq. (5). We call this kind of weighted-CNP *entropic-CNP*. When using this entropic-CNP in LPA, the algorithm is adaptive and more effective, and we call this algorithm LPA-CNP-E.

3.5. Weighted-CNP network

In the steps of the LPA-CNP algorithm described above, we first calculate all the weighted-CNP by Eq. (5) and then use the label propagation strategy. Since all the weighted-CNPs are obtained, we can transform the original network $\boldsymbol{G}$ to a new weighted network $\boldsymbol{G}'$ where the vertices are original vertices and edges connect the pairs of nodes whose weighted-CNP is not zero. The edge's weight is equal to the weighted-CNP of the two nodes it connects. And then the LPA-CNP algorithm is equal to LPA algorithm on this new weighted network $\boldsymbol{G}'$. In light of this idea, we can also extend our weighted-CNP to other algorithms by just running the algorithm on the new weighted network $\boldsymbol{G}'$. We call this new network *weighted-CNP network*. According to this framework, we can also extend weighted-CNP to COPRA [27] which is an algorithm for finding overlapping communities based on LPA.

3.6. Complexity analysis

The *direct propinquity* can be obtained directly from the network with the complexity $O(|E|)$. The *angle propinquity* calculation can be formulated as:

$$\forall v \in V(G), \forall v_i, v_j \in N(v)(i \neq j), \quad Pw(v_i, v_j) \leftarrow Pw(v_i, v_j) + 1.$$

The complexity is $O(|V| \cdot d^2)$ where d is the average degree of the network. The *conjugate propinquity* calculation can be formulated as:

$$\forall (v_s, v_t) \in E(G), \forall v_i,\ v_j \in N(v_s) \cap N(v_t)(i \neq j), \quad Pw(v_i, v_j) \leftarrow Pw(v_i, v_j) + 1.$$

The complexity is $O(|E| \cdot d^2)$. For weighted-CNP, w_1 and w_2 should be settled. If entropic-CNP is used, then the calculation of w_1 and w_2 depends on the column length of Feature-Pair matrix $M_{F\text{-}P}$. The complexity is $O(|V|^2)$. However, the calculation of *feature entropy* depends on the number of non-zero element in corresponding row, i.e. the number of non-zero element of *direct propinquity*, *angle propinquity* and *conjugate propinquity*. Similarly, the complexity of the *feature entropy* for *direct propinquity*, *angle propinquity* and *conjugate propinquity* is $O(|E|)$, $O(|V| \cdot d^2)$ and $O(|E| \cdot d^2)$ respectively. Hence, the complexity of all weighted-CNPs is $O((|V| + |E|) \cdot d^2)$.

Since all the weighted-CNPs are obtained, we have the *weighted-CNP network* $\boldsymbol{G}'$. The LPA-CNP can be viewed as LPA on network $\boldsymbol{G}'$. And as discussed above, the complexity of all weighted-CNPs is $O((|V| + |E|) \cdot d^2)$, i.e. the number of edges in this new weighted network is $O((|V| + |E|) \cdot d^2)$. So the complexity is $O(k \cdot (|V| + |E|) \cdot d^2)$ where k is the iteration times.

In conclusion, the complexity of our LPA-CNP is $O(k \cdot (|V| + |E|) \cdot (|E|/|V|)^2)$. Especially if the network is sparse, which is often true in real social network, complexity can be near linear $O(|V|)$.

4. Experiments and discussion

One common measure for the significance of the identified community structures is *modularity* Q [28].

$$Q = \sum_i (e_{ii} - a_i^2) \tag{13}$$

where e_{ii} is the fraction of edges in community i, and a_i is the fraction of edges that link to vertices in community i. A higher value of Q ($Q \in [-1, 1]$) represents a more significant community structure.

Another commonly used measure is *normalized mutual information (NMI)* [29], which has become a de facto standard for the networks with known communities.

$$NMI(A, B) = \frac{2I(A, B)}{H(A) + H(B)} \tag{14}$$

where A and B denote the two partitions of the network. If the found communities are identical to the real communities, then $NMI(A, B)$ takes its maximum value of 1. If the found communities are totally independent of the real partition, for example when the entire network is classified to be one community, $NMI(A, B) = 0$.

Also variation of information (*VOI*) [30] is a measure of the distance between two clusterings closely related to *NMI*. It can be defined as

$$VOI = H(A|B) + H(B|A) \tag{15}$$

where A and B denote the two partitions. Lower values represent a better correlation between partitions. The value of *VOI* ranges from 0 to $\log |V|$ (V is the set of nodes), so we normalize it by dividing with $\log |V|$.

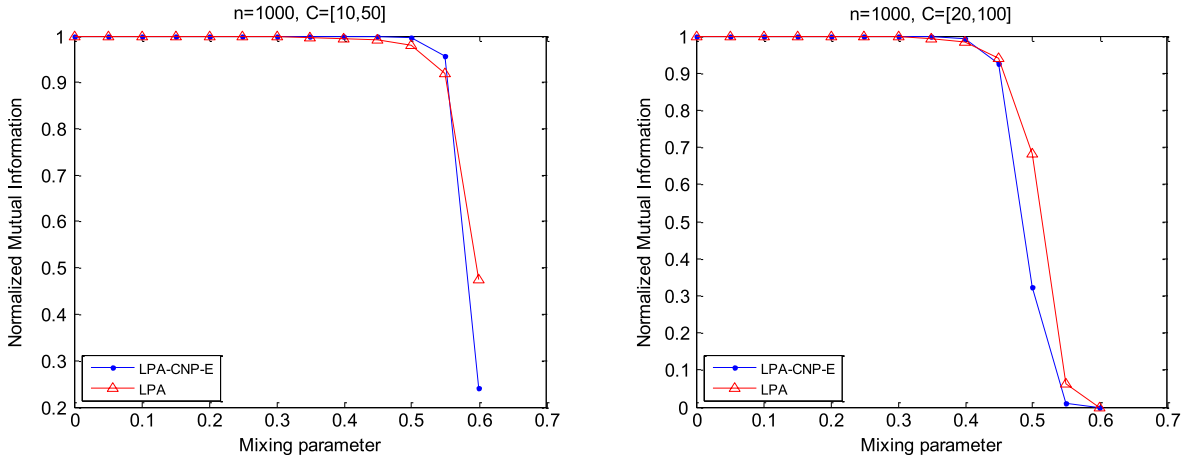

Fig. 3. Average NMI Comparison of LPA-CNP-E and LPA on artificial networks. The number of vertices is 1000 (small scale) and the size of communities varies between [10, 50] (smaller community) and [20, 100] (bigger community) nodes (left, right respectively). We report the averages of NMI over 100 realizations.

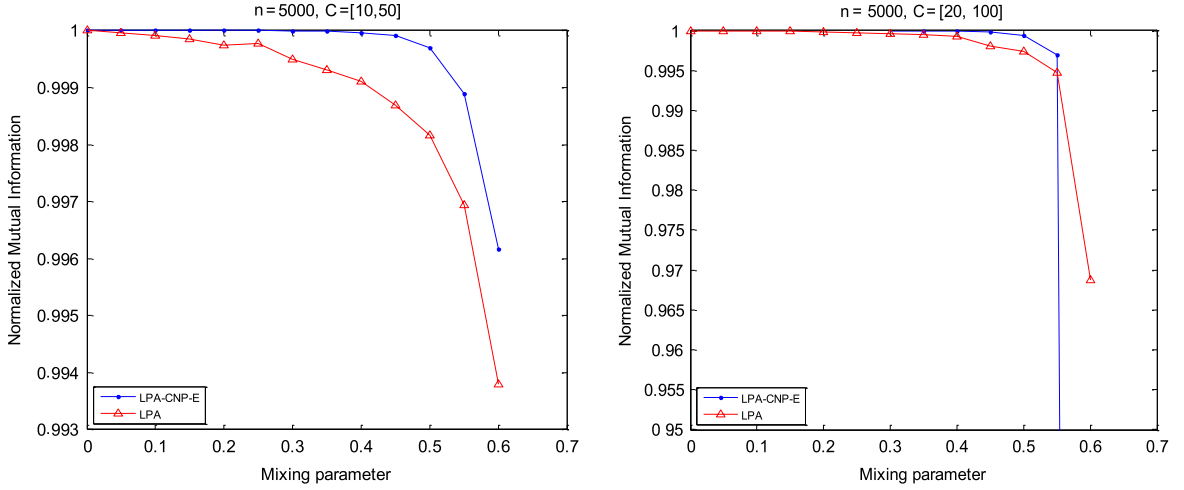

Fig. 4. Average NMI Comparison of LPA-CNP-E and LPA on artificial networks. The number of vertices is 5000 (large scale) and the size of communities varies between [10, 50] (smaller community) and [20, 100] (bigger community) nodes (left, right respectively). We report the averages of NMI over 100 realizations.

4.1. Synthetic networks with planted partition

We first analyze LPA-CNP-E algorithm and LPA on artificial networks with planted partition, LFR benchmark networks [31]. Networks have several parameters. n is the number of vertices; $\langle k\rangle$ and k_{max} are the average and maximum degree which are fixed at 20 and 50 respectively; $\hat{o}_1$ and $\hat{o}_2$ are the exponents of the power-law distribution of vertex degrees and community sizes, which are set to 2 and 1 respectively; c_{min} and c_{max} are the minimum and maximum community size. The significance of community structure is controlled by a mixing parameter $\mu \in [0, 1]$ where smaller values mean clearer community structure. μ is the fraction of internal edges in its community. $\mu < 0.5$ means more inter-community edges than intra-community edges. Results are assessed in terms of NMI, and are shown below.

Figs. 3 and 4 show that when μ is small, i.e. community structure is clearly defined, LPA-CNP-E has better performance than LPA. If we scatter the individual runs (see Figs. 5 and 6), there is actually a significant disparity between LPA-CNP-E and LPA. The NMI of LPA-CNP-E ranges in a narrow interval while LPA ranges in wide interval. The results confirm that LPA-CNP-E is much more robust than LPA.

4.2. Real-world networks with known community structure

LPA-CNP is further analyzed on several real-world networks with community structure (Table 2). All these networks are commonly employed in the community detection literature and all networks are treated as unweighted and undirected. For

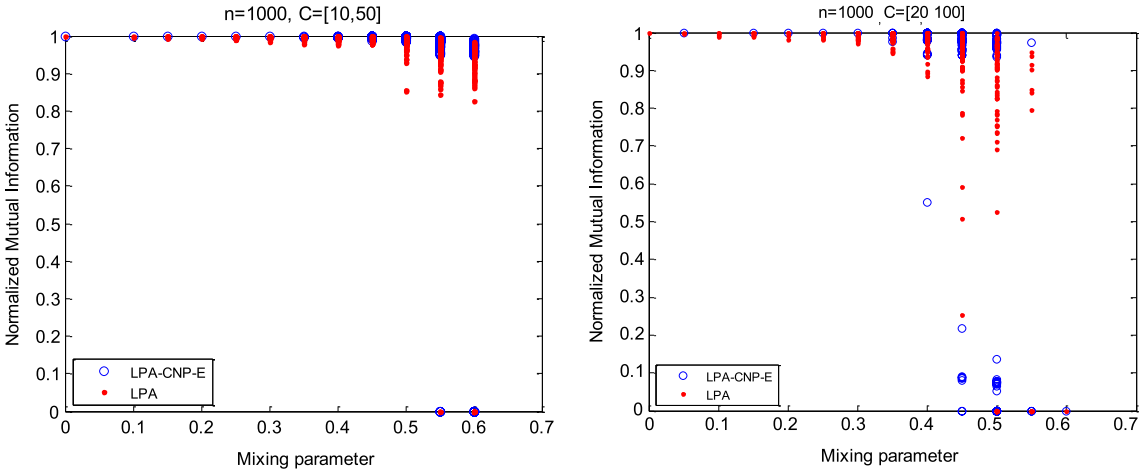

Fig. 5. Individual NMI Comparison of LPA-CNP-E and LPA on artificial networks. The number of vertices is 1000 (small network) and the size of communities varies between [10, 50] (smaller community) and [20, 100] (bigger community) nodes (left, right respectively). We scatter individual runs of NMI over 100 realizations.

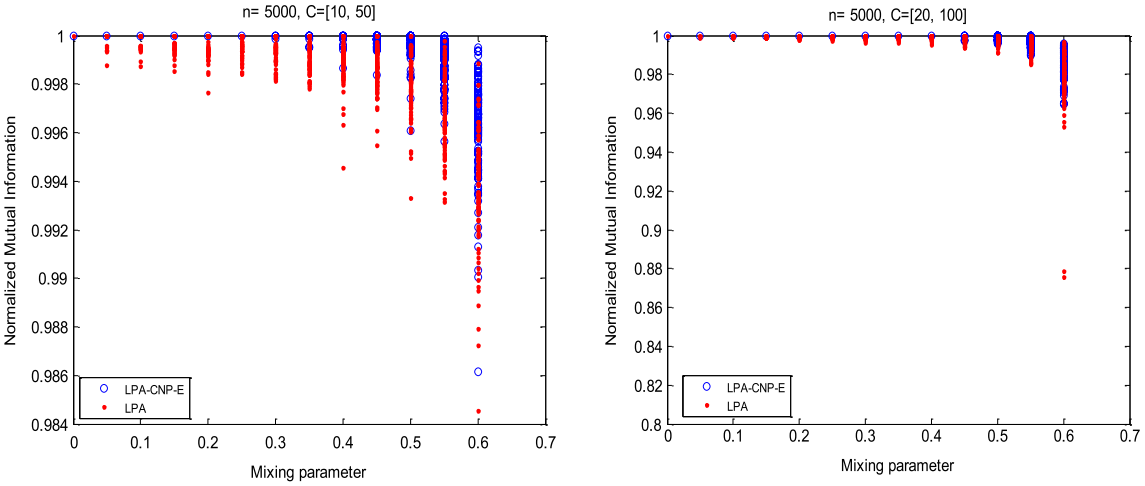

Fig. 6. Individual NMI Comparison of LPA-CNP-E and LPA on artificial networks. The number of vertices is 5000 (large network) and the size of communities varies between [10, 50] (smaller community) and [20, 100] (bigger community) nodes (left, right respectively). We scatter individual runs of NMI over 100 realizations.

Table 2
Real-world networks with community structure.

Network	Description	Vertices	Edges
Karate	Zachary's karate club [32]	34	78
Dolphins	Dolphin social network [33]	62	159
Books	Books about US politics [34]	105	441
Football	American College football [2]	115	613
Blogs	Political blogs [35]	1 490	16 715
Netsci	Network scientists [8]	1 589	2 742
Power	U.S. power grid [36]	4 941	6 594
mat-cond	Condensed matter collaborations [37]	16 726	47 594
astro-ph	Astrophysics collaborations [37]	16 706	121 251

directed or weighted networks, as long as one edge with non-zero weight exists between two nodes, they are treated as connected in new undirected and unweighted networks.

For the LPA-CNP algorithm, two different kinds of weights are adopted in weighted-CNP calculation when performing experiments on real-world networks. One is LPA-CNP-1 with $w_1 = 1$ and $w_2 = 1$. The other one is our LPA-CNP-E. We compare our LPA-CNP with basic LPA. We apply the three algorithms to each network 1000 times and evaluate the

Table 3
Average modularity of LPA-CNP and LPA.

Network	Modularity		
	LPA	LPA-CNP-1	LPA-CNP-E
karate	0.3501	0.2842	0.3027
dolphins	0.4864	0.4576	0.4633
books	0.5035	0.4510	0.4511
football	0.5879	0.6005	0.6006
blogs	0.4176	0.4239	0.4239
netsci	0.8794	0.9314	0.9327
power	0.5952	0.7950	0.8100
mat-cond	0.6750	0.7613	0.7632
astro-ph	0.6274	0.6293	0.6365

Table 4
Average NMI of LPA-CNP and LPA.

Network	NMI		
	LPA	LPA-CNP-1	LPA-CNP-E
karate	0.6400	0.7800	0.8370
dolphins	0.5284	0.7221	0.7313
books	0.5242	0.5706	0.5710
football	0.8894	0.9084	0.9098
blogs	0.3785	0.3911	0.3910
netsci	0.3403	0.3519	0.3523

Table 5
Average pairwise *VOI* of LPA-CNP and LPA.

Network	Pairwise VOI		
	LPA	LPA-CNP-1	LPA-CNP-E
karate	0.1941	0.0704	0.0543
dolphins	0.1814	0.0383	0.0265
books	0.0866	0.0043	0.0040
football	0.0721	0.0136	0.0118
blogs	0.0134	0.0007	0.0008
netsci	0.0305	0.0152	0.0134
power	0.1126	0.1063	0.1015
mat-cond	0.0896	0.0788	0.0763
astro-ph	0.1838	0.0867	0.0863

obtained communities by the above measures. Tables 3–5 show the average performance of the algorithms on the real-world networks. Since we cannot get the real community structure of *power*, *mat-cond* and *astro-ph*, the NMI is ignored.

Table 4 shows LPA-CNP algorithms performance better than LPA. Also LPA-CNP is obviously more robust and stable considering pair-wise *VOI*. As for *modularity* measure, again LPA-CNP show better significance than LPA on most networks except *karate, dolphins* and *books*. We argue that this is an artifact of an intrinsic scale incorporated into the measure of modularity (i.e. *resolution limit* [38,39]).

Looking into LPA-CNP-E on the *karate*, we can find that LPA-CNP-E has only two distinct partitions on the network when running 1000 times. One partition is exactly equal to the real partition and the other shows that entire network is classified into one community. The right partition appears about 837 times in 1000 runs and the other partition appears about 163 times. For LPA, it has 173 distinct partitions meaning weaker robustness. However it only has 2 times classifying the whole network into a community. Though there is more possibility to classify the whole network into one community, LPA-CNP-E gives us a greater possibility of getting an accurate partition. In *karate*, if we already know it should be cut into two or more communities, the result meets the reality exactly.

As for LPA-CNP-E and LPA-CNP-1, the pair-wise *VOI* of LPA-CNP-E is lower than LPA-CNP-1, the *modularity* and *NMI* are always higher. We can know LPA-CNP-E is always more robust and effective.

Also we compare LPA-CNP-E with the balanced propagation algorithm (BPA) mentioned in Section 2. Since nodes, having strong connections with different communities, can prevent BPA from converging [20], we just perform our experiments on networks that converge quickly in BPA. Algorithms are run on each network 1000 times and the results are shown in Table 6.

Compared with BPA, we can find LPA-CNP-E has lower pair-wise *VOI*, which means LPA-CNP has stronger robustness. For *modularity* and *NMI*, LPA-CNP also shows better performance mostly.

Therefore, we can conclude that, in the case of real-world networks, LPA-CNP is indeed more stable, and it also improves the accuracy in most cases especially in large-scale networks.

Table 6
Average modularity of LPA-CNP and LPA.

Network	Modularity		NMI		Pairwise VOI	
	LPA-CNP-E	BPA	LPA-CNP-E	BPA	LPA-CNP-E	BPA
Karate	**0.3027**	0.2856	**0.8370**	0.6338	**0.0543**	0.1413
Dolphins	**0.4633**	0.3789	0.7313	**0.8517**	**0.0265**	0.0684
Books	0.4511	**0.4587**	**0.5710**	0.5622	**0.0040**	0.0740
Football	**0.6006**	0.5994	**0.9098**	0.8796	**0.0118**	0.0682

5. Conclusions

The label propagation algorithm (LPA) focuses on the neighbors of one hop, which is one of the intrinsic reasons for its weak robustness and poor performance. This issue has so far been neglected. This paper points out this issue and introduces weighted coherent neighborhood propinquity (weighted-CNP) to evaluate the probability that a pair of vertices is involved in a same community. The horizon of weighted-CNP is two hops, restricting the calculation in a controllable local scope, while involving enough local topological information. Combining LPA with weighted-CNP, each node is updated by the label with maximum weighted-CNP (LPA-CNP). LPA-CNP extends the horizon of each node to two hops. The weights for weighted-CNP can be adaptive using feature entropy (entropic-CNP). LPA-CNP is applied to both artificial networks with planted partition and several real-world networks with known community structure. The results demonstrate that LPA-CNP is more stable than LPA and more effective in performance. Additionally, we propose a framework to extend the weighted-CNP to other algorithms in detecting community structure. In the future, we will work on other algorithms with weighted-CNP to discuss whether weighted-CNP can improve the performance and how it affects the original algorithms.

Acknowledgments

This work is supported by National Science Foundation of China (61271316, 61071152, 61071081), 973 Program of China (2010CB731403, 2010CB731406) and Chinese National "Twelfth Five-Year" Plan for Science & Technology Support (2012BAH38 B04).

References

[1] M.E.J. Newman, The structure and function of complex networks, SIAM Rev. 45 (2) (2003) 167–256.
[2] M. Girvan, M.E.J. Newman, Community structure in social and biological networks, Proc. Natl. Acad. Sci. USA 99 (12) (2002) 7821–7826.
[3] S. Wasserman, K. Faust, Social Network Analysis, Cambridge University Press, Cambridge, 1994.
[4] G.W. Flake, S.R. Lawrence, C.L. Giles, F.M. Coetzee, Self-organization and identification of web communities, IEEE Trans. Comput. 35 (3) (2002) 66–71.
[5] B.W. Kernighan, S. Lin, An efficient heuristic procedure for partitioning graphs, Bell. Syst. Tech. J. 49 (1970) 291–307.
[6] L. Donetti, M.A. Muñoz, Detecting network communities: a new systematic and efficient algorithm, J. Stat. Mech.: Theory Exp. (10) (2004) 10012.
[7] A. Capocci, V.D.P. Servedio, G. Caldarelli, F. Colaiori, Detecting communities in large networks, Phys. A 352 (2–4) (2005) 669–676.
[8] M. Fiedler, Algebraic connectivity of graphs, Czech Math. J. 23 (98) (1973) 298–305.
[9] M.E.J. Newman, Fast algorithm for detecting community structure in networks, Phys. Rev. E 69 (2004) 066133.
[10] Roger Guimera, Marta Sales-Pardo, A. Luis, Nunes Amaral, Modularity from fluctuations in random graphs and complex networks, Phys. Rev. 70 (2004) 025101.
[11] J. Duch, A. Arenas, Community detection in complex networks using extremal optimization, Phys. Rev. E 72 (2005) 027104.
[12] M. Newman, Finding community structure in networks using the eigenvectors of matrices, Phys. Rev. E 74 (3) (2006) id. 036104.
[13] G. Palla, I. Derényi, I. Farkas, T. Vicsek, Uncovering the overlapping community structure of complex networks in nature and society, Nature 435 (2005) 814–818.
[14] M. Rosvall, C.T. Bergstrom, An information-theoretic framework for resolving community structure in complex networks, Proc. Natl. Acad. Sci. 104 (18) (2007).
[15] M. Rosvall, C.T. Bergstrom, Maps of random walks on complex networks reveal community structure, Proc. Natl. Acad. Sci. 105 (4) (2008).
[16] P. Ronhovde, Z. Nussinov, Multiresolution community detection for mega scale networks by information-based replica correlations, Phys. Rev. E 80 (2009).
[17] U.N. Raghavan, R. Albert, S. Kumara, Near linear time algorithm to detect community structures in large-scale networks, Phys. Rev. E 76 (2007) 036106.
[18] L. Subelj, M. Bajec, Unfolding communities in large complex networks: Combining defensive and offensive label propagation for core extraction, Phys. Rev. E 83 (2011) 036103.
[19] Y.Z. Zhang, J.Y. Wang, Y. Wang, L.Z. Zhou, Parallel community detection on large networks with propinquity dynamics, 15th ACM SIGKDD international conference on Knowledge discovery and data mining, New York, 2009, pp. 997–1006.
[20] L. Šubelj, M. Bajec, Robust network community detection using balanced propagation, Eur. Phys. J. B 81 (3) (2011) 353–362.
[21] L.X.Y. Leung, P. Hui, P. Lio, J. Crowcroft, Towards real-time community detection in large networks, Phys. Rev. E 79 (6) (2009) 066107.
[22] C.E. Shannon, A mathematical theory of communication, ACM SIGMOBILE Mob. Comput. Commun. Rev. 5 (2011) 3–55.
[23] R. Martin, Information horizons in a complex world, Ph.D. dissertation, Dept. Phys., Umeâ Univ., Umeå, 2006.
[24] J. Pei, D.X. Jiang, A.D. Zhang, On mining cross-graph quasi-cliques, The eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, New York, 2005, pp. 228–238.
[25] Z. Zeng, J. Wang, L. Zhou, G. Karypis, Out-of-core coherent closed quasi-clique mining from large dense graph databases, ACM Trans. Database Syst. 32 (13) (2007).
[26] Y. Liu, L. Liu, J.Y. Luo, The adaptive method for closely communicating community detection based on ant colony clustering, International Conference on Multimedia Information Networking and Security, 2010, pp. 250–254.
[27] Steve Gregory, Finding overlapping communities in networks by label propagation, New J Phys. 12 (2010) 103018.
[28] M.E.J. Newman, M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E 69 (2004) 026113.
[29] L. Danon, A. Diaz-Guilera, J. Duch, A. Arenas, Comparing community structure identification, J. Stat. Mech. 09 (2005).
[30] M. Meila, Comparing clusterings—an information based distance, J. Multivariate Anal. 98 (2007) 873–895.

[31] A. Lancichinetti, S. Fortunato, F. Radicchi, Benchmark graphs for testing community detection algorithms, Phys. Rev. E 78 (4) (2008) 046110.
[32] W.W. Zachary, An information flow model for conflict and fission in small groups, J. Anthropol. Res. 33 (4) (1977) 452–473.
[33] D. Lusseau, K. Schneider, O.J. Boisseau, P. Haase, E. Slooten, S.M. Dawson, The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations, Behav. Ecol. Sociobiol. 54 (2003) 396–405.
[34] V. Krebs, A network of co-purcheased books about U.S. politics, 2008 [Online]. Available: http://www.orgnet.com.
[35] L.A. Adamic, N. Glance, The political blogosphere and the 2004 US Election, in: Proceedings of the WWW-2005 Workshop on the Weblogging Ecosystem, 2005.
[36] D.J. Watts, S.H. Strogatz, Collective dynamics of small-world networks, Nature 393 (1998) 440–442.
[37] M.E.J. Newman, The structure of scientific collaboration networks, Proc. Natl. Acad. Sci. USA 98 (2001) 404–409.
[38] S. Fortunato, M. Barthelemy, Resolution limit in community detection, Proc. Natl. Acad. Sci. USA 104 (1) (2007) 36–41.
[39] B.H. Good, Y.A. de Montjoye, A. Clauset, The performance of modularity maximization in practical contexts, Phys. Rev. E 81 (4) (2010) 046106.