Knowledge-Based Systems 206 (2020) 106363

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

GLLPA: A Graph Layout based Label Propagation Algorithm for community detection

Yun Zhang [a], Yongguo Liu [a,*], Rongjiang Jin [b], Jing Tao [c], Lidian Chen [c], Xindong Wu [d]

[a] Knowledge and Data Engineering Laboratory of Chinese Medicine, School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu 610054, China
[b] College of Health Preservation and Rehabilitation, Chengdu University of Traditional Chinese Medicine, Chengdu 610075, China
[c] College of Rehabilitation Medicine, Fujian University of Traditional Chinese Medicine, Fuzhou 350122, China
[d] Mininglamp Academy of Sciences, Mininglamp Technology, Beijing 100190, China

ARTICLE INFO

Article history:
Received 26 February 2020
Received in revised form 24 June 2020
Accepted 3 August 2020
Available online 7 August 2020

Keywords:
Community detection
Label propagation
Graph layout
Node attraction
Node influence
Label influence

ABSTRACT

Community is an important property of networks. Recently, label propagation based community detection algorithms develop rapidly, since they can discover communities with high efficiency. However, the results of most of them are inaccurate and unstable because the node order of label updating and the mechanism of label propagation are random. In this paper, a new label propagation algorithm, Graph Layout based Label Propagation Algorithm (GLLPA), is proposed to reveal communities in networks, which aims at detecting accurate communities and improving stability by exploiting multiple graph layout information. Firstly, GLLPA draws networks to compact layout based on the force-directed methods with (a,r)-energy model, then a label initialization strategy is proposed to assign the nodes locating in a position with the same label. Secondly, GLLPA begins to draw networks to uniform layout and conduct community detection simultaneously, in which we design node influence and label influence based on node attraction in the uniform layout to handle the instability problem and enhance its accuracy and efficiency. Experimental results on 16 synthetic and 15 real-world networks demonstrate that the proposed method outperforms state-of-the-art algorithms in most networks.

1. Introduction

Complex systems can be modeled as complex networks, in which nodes represent real entities and edges represent the interactions among entities [1]. Real-world networks often consist of functional units that manifest in the forms of subnetworks or communities with nodes more tightly connected with respect to the rest of networks [2,3]. Community detection can be informally described as a problem of finding such communities in networks, which aims at assigning community labels to nodes such that the nodes in the same community share higher similarity than the nodes in different communities [4,5]. Nodes own different influences defined as node weight, for example, the node with large degree owns high influence [6,7].

To detect communities, researchers designed various types of algorithms, such as modularity maximization, feature mapping (spectral clustering, Nonnegative Matrix Factorization (NMF) and embedding-based methods) and label propagation. Here we introduce some of them briefly.

Modularity-maximization based approaches consider community detection problem as an optimization problem to maximize modularity for discovering communities [2,8–10]. Modularity is a measurement of community quality by considering the degree distribution of nodes [8]. These methods mainly design different hierarchical clustering strategies to divide networks, which are time-consuming [11]. Meanwhile, they suffer from resolution limitation that prevents them from detecting communities which are comparatively small with respect to the whole network [9]. But these methods can detect the number of communities automatically and find communities with high internal density [10].

Feature mapping based community detection models mainly map nodes to low-dimensional space and use traditional clustering algorithms such as k-means [12] and Gaussian Mixture Model (GMM) [13] to discover communities. The proximity among connectivity nodes should be kept in new space. The representative models mainly contain spectral clustering [14,15], NMF [16,17] and embedding-based models [18,19].

- Spectral clustering based methods compute Laplace matrix based on the similarity or adjacency matrix of networks [14,15]. Laplace matrix is defined as $\boldsymbol{L} = \boldsymbol{K} - \boldsymbol{A}$, where $\boldsymbol{A}$ is the similarity or adjacency matrix and $\boldsymbol{K}$ is the degree

* Corresponding author.
E-mail address: liuyg@uestc.edu.cn (Y. Liu).

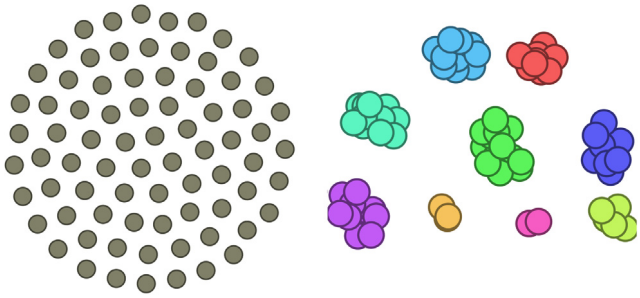
https://doi.org/10.1016/j.knosys.2020.106363
0950-7051/

matrix of a network. Then the eigenvalue and eigenvector of Laplace matrix are calculated to form the eigenvector matrix corresponding to the top-k eigenvalue, where k is the same as the number of communities. Spectral clustering methods only need the similarity or adjacency matrix of networks, so they are effective for detecting communities in sparse networks. However, when dealing with large scale networks, the matrix space will be large, then the computation is time-consuming. Meanwhile, they need to set the number of communities manually [15].

- NMF-based community detection methods approximately factorize the similarity or adjacency matrix $\boldsymbol{A}$ of networks into two non-negative factor matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ (i.e., $\boldsymbol{A} \approx \boldsymbol{UV}$ ($\boldsymbol{U} \geq 0$, $\boldsymbol{V} \geq 0$)) [16,17]. Then basis vector $\boldsymbol{U}$ with fewer dimensions can be used to represent entire data space and decomposition results $\boldsymbol{U}$ and $\boldsymbol{V}$ have good interpretability. For example, each column of factor matrix $\boldsymbol{V}$ can be explained as the membership of a node belonging to different communities and factor matrix $\boldsymbol{U}$ can be considered as the mapping between the original network and the community membership space [17]. However, when dealing with large scale networks, the matrix space also will be large, which is similar with spectral clustering based methods, then the factorization is complicated. In general, matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ are randomly initialized, which will result in unstable results and reduce convergence speed [16].
- Recently, embedding-based methods are popular for community detection [18,19], which represent nodes as low dimensional features by network embedding (graph embedding) methods [20], then community detection problem can be transformed into node vector clustering problem [18]. Nodes with similar features have similar vector representations, then clustering algorithms can be used to discover node membership to form communities. Based on node embeddings, some methods also learn community embeddings as multivariate distributions to improve community quality [18]. Embedding-based community detection methods can capture potential network features and high order information, such as hierarchical structure. However, they need high computational overhead [19].

Label propagation based algorithms mainly utilize the neighbor node information of each node to determine its label, which do not need any prior knowledge of communities [21–23]. Raghavan et al. [24] first introduced label propagation into community detection and proposed Label Propagation Algorithm (LPA) that simply updates the label of nodes by the most label of neighbor nodes. LPA has nearly linear time complexity and receives great attention. However, the accuracy and stability of community results are sensitive to the node order of label updating and the order of label selection.

Above algorithms have been applied to find communities and received effective results. In this paper, we focus on label propagation based algorithms for community detection due to their simplicity and near-linear time complexity [21,22]. Meanwhile, we do not need to set the number of communities because they can detect community number automatically [24]. In fact, we cannot know the number of communities of most real-world networks in advance, then label propagation based algorithms are suitable for discovering communities in real-world networks [23,25,26]. Raghavan et al. [24] firstly proposed LPA, which has linear time complexity. However, LPA has some problems: (1) it cannot discover overlapping communities in networks because it only assigns a node with one label; (2) its accuracy is relatively weak in some networks, which probably because it considers all nodes and labels with the same weight; (3) the node order of label updating is random, which also may reduce its accuracy and result in unstable community results.



(a) Uniform layout (b) Compact layout

Fig. 1. Examples of network layout. (The edges are omitted).

For discovering overlapping communities, Community Overlap PRopagation Algorithm (COPRA) [27] and Speaker–Listener Label Propagation Algorithm (SLPA) [28] are proposed, which allocate several labels with belonging coefficients (i.e., label weight) to a node for identifying the membership that a node belongs to multiple communities. However, COPRA and SLPA also have some weaknesses: (1) they also do not take into account node weight, which may lead to weak accuracy; (2) they also select nodes in random order to update labels and spread one of multiple labels with the same belonging coefficient of a node to other nodes randomly, then they do not deal with the instability problem well.

For improving the instability problem, many node weight and label weight based label propagation algorithms, such as NIBLPA [29], DLPA$^+$ [30], WLPA [31], LPA_NI [32] and NGLPA [21] are proposed to regulate the node order of label updating and the order of label selection. Their results illustrate that considering the weight of nodes and labels can enhance the accuracy and stability. However, most of their results are also inaccurate and unstable. The nodes or labels may have the same weight, then the methods will randomly choose one of them to update or propagate, which leads to inaccurate communities and the instability problem. Meanwhile, existing label propagation based algorithms initialize each node owns unique label, which may increase iteration times to reduce the number of labels to the number of communities.

In this paper, a new label propagation algorithm, Graph Layout based Label Propagation Algorithm (GLLPA), is proposed for detecting communities in networks for enhancing accuracy and efficiency and improving the instability problem by determining the node order of label updating and the rule of label launch and label acceptation, which utilizes graph layout information to assist model discover communities.

Graph layout, also known as network layout, is a procedure of deriving a pictorial representation of the nodes and edges of a network [33,34]. Many graph layout methods have been proposed, in which Fruchterman–Reingold algorithm (FR), an effective force-directed layout algorithm, regards nodes as particles and considers the attractive and repulsive forces among them to exhibit nodes and their relations [35,36]. Meanwhile, it is known that the force is the negative gradient of energy [33], so energy models also can represent the force systems to draw networks [37]. Among them, (a, r)-energy model is used frequently, which can represent multiple layout algorithms by adjusting the values of a and r [33]. Visualizing networks can help us better understand the structures of real-world networks [38], which can embody real objects and their relationships. For example, as shown in Fig. 1(a), the uniform layout of a given network can reflect its overall situation [35,36]. As shown in Fig. 1(b), the

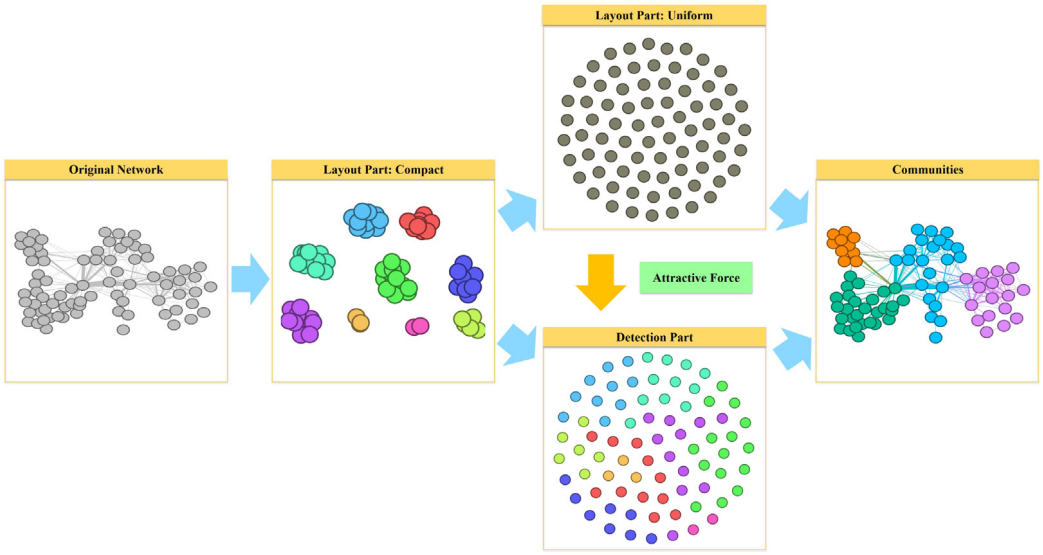


Fig. 2. The procedure of GLLPA. Firstly, GLLPA draws networks to compact layout, then label initialization strategy assigns the nodes in a position with identical label. Secondly, GLLPA begins to draw networks to uniform layout and conduct community detection simultaneously. The node influence and label influence used in community detection part change according to the variation of layout. Finally, when the community detection part reach termination condition, layout part and detection part stop and GLLPA assigns nodes with the same label to the same community.

compact layout of networks can reflect the proximity relations among nodes. Nodes with large attraction will gather together in this layout to form groups, and some nodes in the same true communities tend to collapse into a single position [33].

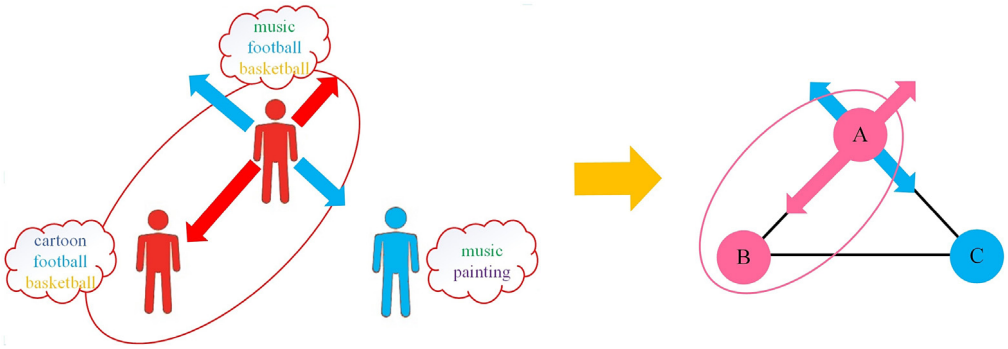
Based on the force-directed layout algorithms with (a, r)-energy model [39], we propose GLLPA to discover communities for enhancing accuracy and efficiency and improving the instability problem. We mainly utilize graph layout information, which is based on the previous works (compact layout [33] and uniform layout [35]), to assist the community detection to discover communities in networks. The procedure of GLLPA is shown in Fig. 2. Concretely, GLLPA contains two parts, network layout and community detection. GLLPA first draws the networks to compact layout, then plots the networks to uniform layout and conduct community detection simultaneously. Based on the results of compact layout, a new label initialization strategy is proposed to assign the nodes in a position with the same label for community detection. By utilizing the attractive force in the process of drawing networks to uniform layout, we design node influence and label influence to relatively accurately measure their weight to enhance performance and improve the instability problem.

Inspired from three bio-examples, human society (interest groups), ant life (hierarchy) and bat population (echolocation behavior), respectively, we consider the attraction among nodes, the influence of nodes and the influence of labels when they are launched from a node to other nodes to handle the current existing problems of label propagation based algorithms. Meanwhile, we propose a new label initialization strategy to assign the nodes in a position with the same label.

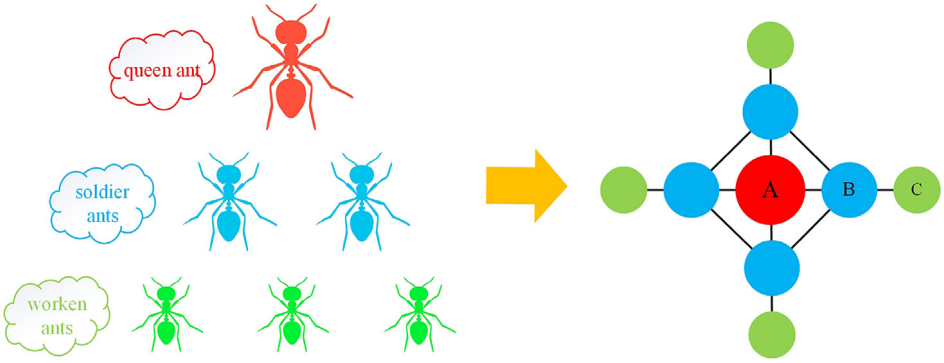
- **Inspiration 1 Interest groups of people.** Inspired from the interest groups of people, we propose a new label propagation strategy to assign nodes locating in the same position with the same label in the compact layout and define node attraction to reflect the proximity among nodes.
- **Inspiration 2 Hierarchy of ants.** Inspired from the hierarchy of ants, we design node influence to measure the weight of nodes according to the degree and centrality of nodes and their neighbors based on node attraction.
- **Inspiration 3 Echolocation behavior of bats.** Inspired from the echolocation behavior of bats, we define label influence to evaluate the weight of labels when they are launched from a node to another node.

In real life, if people acquainted mutually have more common interests, then they often participate in the same group [40,41]. Meanwhile, if they have different interests, then they may have different perspectives about different interests [40,41]. In other words, if people communicate with others closely and have close relationship (large attraction), then they belong to the same group possibly. On the other hand, there is repulsion to a certain extent among them for incompletely identical hobbies. As shown in Fig. 3(a), person A has three interests (music, football and basketball), person B has three interests (cartoon, football and basketball) and person C has two hobbies (music and painting), then persons A and B tend to belong to the same community for more common interests. Because persons A and B also have different interests, they may have different ideas about music and cartoon. Thus, there also exists repulsion between persons A and B, but the attraction is larger than the repulsion between them for more common interests, then they belong to the same community possibly. In order to mimic the attraction and repulsion among people, we define node attraction and node repulsion in networks by the attractive and repellent forces among nodes in the force-directed layout methods [33,34]. Inspired from the interest groups of people, the higher the node attraction between a node and its neighbors, the higher probability the two nodes belong to the same community. If nodes are gathered in one position in the compact layout as shown in Fig. 1(b) (i.e., their node attraction is large), then they belong to the same community possibly [33]. Thus, we propose a new label initialization strategy in community detection to assign nodes locating in the same position with the same label in the compact layout for increasing accuracy and accelerating convergence.

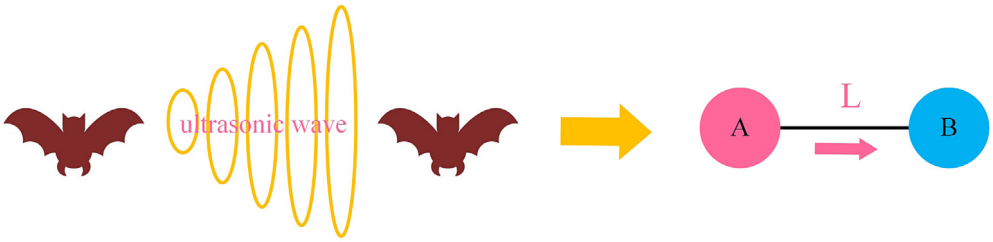
For social animals that live in groups, such as ants, animals having different roles own different influence, that is, some individuals are leaders and others are followers [42,43]. As shown in Fig. 3(b), ant group has hierarchy. In other words, the leader (e.g., queen ant) has the highest influence and others (e.g., worker ants, soldier ants) have different influence according to their roles in animal groups [44]. On the other hand, the influence of an animal also comes from its followers [42,45,46]. For example,



(a) Interest groups of people → Label initialization and node attraction



(b) Hierarchy of ants → Node influence



(c) Echolocation behavior of bats → Label influence

Fig. 3. The inspirations from three bio-examples.

under the condition that animals A and B have the same influence themselves, if the followers of animal A own stronger influence than those of animal B, then animal A is often viewed to own larger influence than animal B [42,44]. Thus, we consider that the total influence of an animal consists of personal influence and followers' influence, which increases with the promotion of roles in its group [42,43]. Meanwhile, the animal with large influence affects the one with small influence in a high possibility [43,44]. For example, the leader can order other animals, while the followers cannot command the leader. By modeling animal group as a social network, we can infer that the node influence is made up of personal influence and neighbor influence. Then there is a high possibility that the node with high influence will spread labels to the nodes with low influence. A node has large influence itself if it owns large degree or locates at the centric position of networks [6,7]. In GLLPA, node influence is proposed by the degree and centrality of a node and its neighbors to denote its weight in networks. Existing node weight based label propagation methods consider that the weight of nodes in networks is changeless [5,21,30,32,47]. In this paper, inspired by the promotion of animal roles, we define node influence, which can vary dynamically according to the change of layout. When the layout of networks changes, we use the node attraction to define the centrality of a node to measure node influence. Node influence can reflect the weight of nodes relatively comprehensively to relieve the problem that nodes may have the same weight for increasing accuracy. In addition, we fix the node order of label updating according to node influence in one iteration so as to handle the randomness problem.

In the groups of bats, as shown in Fig. 3(c), when a bat sends information to others by echolocation behavior during the process of ultrasonic wave transmission, the intensity of the information accepted by other bats depends on the power of the sender, propagation distance and the importance of information [48,49]. We model bats as nodes and their interactions

as edges, then we can gain a network, thus, we can consider the propagating information as label. Existing label propagation based algorithms make the weight of labels of a node (i.e., belonging coefficient) invariable during label propagation process [21,31,30,32,47]. Like the echolocation behavior of bats, when the labels arrive at other nodes, the weight of labels in networks should change according to the node influence of launcher, the distances among nodes and the belonging coefficients of labels. Thus, we design label influence to reflect the weight of labels when they arrive at a node according to above three factors, which can reflect the weight of labels relatively accurately to overcome the problem that labels may have the same weight in the spreading process for enhancing stability. GLLPA filters the labels with low influence to discover accurate communities and accelerate convergence.

Experimental results on synthetic and real-world networks show that the proposed algorithm outperforms state-of-the-art algorithms in most networks. The main contributions of this paper can be summarized as follows.

- Based on the compact layout of networks, we propose a new label initialization strategy to assign nodes locating in the same position with the same label to reduce iteration time.
- Based on the attractive force in uniform layout of networks, inspired from bio-examples, we design node influence and label influence to relatively accurately measure their weight for increasing accuracy and handling the instability problem.
- Graph layout based label propagation algorithm GLLPA is proposed for discovering communities in networks with the new label initialization strategy and node attraction, node influence and label influence.
- We evaluate the effectiveness of GLLPA on 16 synthetic and 15 real-world networks.

2. Related work

Community detection methods based on label propagation are efficient for discovering communities. Here, we introduce some methods that use label propagation technology. Then we introduce some other types of community detection algorithms.

2.1. Label propagation based methods

In this section, we first introduce two algorithms COPRA and SLPA, which are used for detecting overlapping communities in networks. Then we introduce several methods to discover communities by (1) considering the weight of node itself and the weight of labels (WLPA and LPA_NI) and (2) considering the weight of nodes and their neighbors and the weight of labels (DLPA, DLPA$^+$ and NGLPA). After that, LinkLPA is introduced that considers edge information to find communities by transforming node partition problem into edge partition problem. Finally, we introduce two algorithms MWLP and GLPA by transforming the original unweighted networks to weighted networks to uncover communities.

COPRA [27] and SLPA [28] are proposed for discover overlapping communities in networks. For the former, each node owns a label set and the labels have belonging coefficients which reflect their membership among communities and are updated by averaging the belonging coefficients over all labels in neighbors. For the latter, nodes have memory spaces to store received labels as historical information, then nodes are allocated to communities according to the frequency of labels in memory spaces.

To better discover communities, researchers proposed many label propagation based algorithms to handle the instability problem by considering the weight of nodes and labels. On the one hand, researchers considered the weight of node itself and labels. Tong et al. [31] proposed Weighted Label Propagation Algorithm (WLPA) to define degree as node weight and consider the weight of labels defined by the ratio of nodes' degree to the average degree of networks. WLPA propagates the labels with large weight to handle the randomness. Zhang et al. [32] proposed label propagation algorithm LPA_NI based on node and label weight. LPA_NI defines node weight by the priori importance of nodes calculating by a Bayesian network from expert knowledge. In addition, label weight is calculated by the weight and degree of the nodes to determine the order of label choosing. LPA_NI determines the node order of label updating according to node weight and propagates the labels with large weight to other nodes. On the other hand, researchers considered the weight of nodes and their neighbors and label weight. Sun et al. [50] proposed Dominant Label Propagation Algorithm (DLPA) to consider the weight of neighbor nodes. The confidence of the neighbors to a node is defined to evaluate the weight of neighbors and calculate dominant labels. As a result, DLPA propagates dominant labels to improve the instability. However, DLPA also updates the labels of nodes in a random order. Liu et al. [30] proposed DLPA$^+$ and introduced the confidence variance of nodes to improve DLPA's instability by updating nodes according to their confidence variance. Shen and Ma [21] proposed a Node Gravitation based Label Propagation Algorithm (NGLPA), in which labels are propagated according to node gravitation defined by node weight and node similarity. Different with above methods, NGLPA defines node weight by LeaderRank [51] that computes the relations among nodes and its neighbors iteratively. Beside taking into account node information, some methods are proposed to detect communities by considering edge information. Sun et al. [25] proposed Link-based Label Propagation Algorithm (LinkLPA), which transforms node partition problem into edge partition problem, assigns the edges with unique labels and employs LPA with preference on edges instead of nodes to detect communities. Finally, LinkLPA takes a post-processing step to refine the detected communities by eliminating the over-overlapping nodes.

In addition, some algorithms transform original unweighted networks to weighted networks and design some strategies to uncover communities. Li et al. [23] proposed Motif-aware Weighted Label Propagation (MWLP) for community detection. First, MWLP focuses on the triangles within the network and identifies the motif of interest (locally meaningful pattern), then the motif-based hyper graph is constructed. Secondly, a weighted network is built by unifying both the higher-order structure (hyper graph) and the original lower-order structure. Then MWLP updates node labels by both the number and strength of edges in the weighted network. Yang et al. [26] proposed Graph-based Label Propagation Algorithm (GLPA) to detect communities incorporating the node similarity and connectivity information. First, GLPA constructs a label propagation graph by choosing node and its most similar neighbor node, and then calculates its connected components. Secondly, GLPA constructs a weighted graph by treating each connected component as a super-node. Finally, GLPA computes the merging factor for each node in the weighted graph and merge super nodes with higher merging factor to its most similar node iteratively to reach the maximum complementary entropy to detect communities.

However, above methods considers that the weight of nodes and labels is invariable, we argue that the weight should change inspired from real world. In this paper, based on node attraction, we propose node influence varying dynamically according to the change of layout to measure the weight of nodes. And we design label influence to measure the weight of labels that may vary when a label is launched to other nodes.

2.2. Non-label propagation based methods

In this section, we introduce some other types of community detection methods.

Haq et al. [9] proposed a community detection approach based on weighted modularity to improve the resolution limit of the modularity function by incorporating a weight term in the modularity formulation that measures how strong a community is in terms of its conductance, i.e., the ratio of the edges within a community to the maximum number of possible edges.

Li et al. [15] proposed an overlapping community detection algorithm based on the node convergence degree, which is defined as the combination of attribute convergence degree and structure convergence degree. An improved PageRank algorithm is used to get the importance of each node in the global network and the information of local network is used to measure the structure convergence degree. The overlapping communities are identified by spectral clustering based on the node convergence degree.

Ye et al. [17] proposed Deep Autoencoder-like NMF (DANMF) algorithm for detecting communities, which approximately factorizes the similarity or adjacency matrix $\boldsymbol{A}$ of networks into multiple factor matrices $\boldsymbol{U_1}, \ldots, \boldsymbol{U_p}$ and $\boldsymbol{V_1}, \ldots, \boldsymbol{V_p}$, where p is the number of layers. DANMF is based on an architecture that is similar to deep autoencoder to learn the hierarchical mappings between the original network and the community assignment with implicit low-to-high level hidden features.

Cavallari et al. [18] proposed a Community Embedding algorithm (ComE) that jointly solves community embedding, community detection and node embedding together. On the one hand, node embedding can help improve community detection, which outputs good communities for fitting better community embedding. On the other hand, community embedding can be used to optimize the node embedding by introducing a community-aware high-order proximity.

3. Preliminaries

Let $G = (V, E)$ be a given undirected and unweighted network, where $V = \{v_1, \ldots, v_i, \ldots, v_n\}$ denotes the set of nodes, $E = \{e_1, \ldots, e_i, \ldots, e_m\}$ denotes the set of edges, and n and m are the number of nodes and edges, respectively. The degree of node v_i is denoted as k_{v_i} and the neighbor node set of node v_i is denoted as $N(v_i) = \{v_j | e_{v_i v_j} \in E\}$. The labels of node v_i are denoted as $M(v_i) = \{(l_1^{v_i}, c_1^{v_i}), \ldots, (l_j^{v_i}, c_j^{v_i}), \ldots, (l_H^{v_i}, c_H^{v_i})\}$, where label $l_j^{v_i}$ with belonging coefficient $c_j^{v_i}$ is the jth label of node v_i, $\sum_{j=1}^{H} c_j^{v_i} = 1$ and H is the number of labels. The notations used in GLLPA are shown in Table 1.

3.1. Node attraction

For network G, layout $p : V \rightarrow S$ is a function that maps nodes in V to a set of corresponding positions in layout space S, e.g., $S = R^2$. Node attraction reflects the attractive function between nodes v_i and v_j in layout p [35], which is defined as

$$F^a(v_i, v_j) = \left\| p(v_i) - p(v_j) \right\|^a \overrightarrow{p(v_i)p(v_j)}, \tag{1}$$

where $p(v_i)$ is the position of node v_i in layout p, $\left\| p(v_i) - p(v_j) \right\|$ is the distance between nodes v_i and v_j, $\overrightarrow{p(v_i)p(v_j)}$ is a unit-vector pointing from $p(v_i)$ to $p(v_j)$ and a is the attractive parameter to adjust the influence of distances on node attraction. If the attractive forces among nodes increases, then these nodes tend to gather together. With the change of network layout, node attraction varies dynamically until the layout is steady.

3.2. Node repellent

Node repellent reflects the repulsive function between nodes v_i and v_j in layout p [35], which is defined as

$$F^r(v_i, v_j) = k_{v_i} k_{v_j} \left\| p(v_i) - p(v_j) \right\|^r \overrightarrow{p(v_i)p(v_j)}, \tag{2}$$

where r is the repulsive parameter to adjust the effect of distances on node repellent. If the repulsive forces among nodes increases, then these nodes tend to disperse. In similar manner, with the change of network layout, node repellent varies dynamically until the layout is steady. For node attraction and node repellent, parameters a and r should meet the condition that $a > r$ to ensure that the attractive forces between connected nodes grows faster than the repulsive forces, and thus prevents infinite distances except between unconnected components [39].

3.3. Layout energy

GLLPA draws networks based on a general force-directed layout framework, whose energy model is called (a, r)-energy model [39]. By exploiting the fact that force is the negative gradient of energy (i.e., the energy is the negative integrals of force) [39], we use the energy model and consider the interactions among adjacent nodes to calculate energy $\varepsilon(p|G)$ of layout p for network G, which is defined as

$$\begin{aligned} \varepsilon(p|G) &= -\sum_{e_{v_i v_j} \in E} k_{v_i} k_{v_j} \frac{\|p(v_i) - p(v_j)\|^{r+1}}{r+1} \\ &\quad - \left(-\sum_{e_{v_i v_j} \in E} \frac{\|p(v_i) - p(v_j)\|^{a+1}}{a+1}\right) \\ &= \sum_{e_{v_i v_j} \in E} \frac{\|p(v_i) - p(v_j)\|^{a+1}}{a+1} \\ &\quad - \sum_{e_{v_i v_j} \in E} k_{v_i} k_{v_j} \frac{\|p(v_i) - p(v_j)\|^{r+1}}{r+1}. \end{aligned} \tag{3}$$

The first term can be seen as the sum of the negative integrals of node attraction (Eq. (1)), which represents the sum of the energy produced by the attractive forces among adjacent nodes. The second term can be seen as the sum of the negative integrals of node repellent (Eq. (2)), which denotes the sum of the energy produced by the repulsive forces among adjacent nodes. Because the direction of node attraction and node repellent is opposite, then we compute their difference value of energies (the energy of node repellent minus the energy of node attraction) to obtain Eq. (3) for representing the whole energy of network layout. Graph layout can be seen as an optimization problem with the objective function of Eq. (3) over all possible layout p. The layout has different characteristics depending on the values of parameters a and r [33,39].

In order to obtain the compact layout of networks, the nodes of the same community should collapse into a single position in layout area ideally [33], thus, the attractive forces among nodes of the same community should be stronger than their repulsive forces to get nodes closer to each other. Meanwhile, the total stress condition of layout should become equilibrium state [33, 39]. Based on the compact layout, we propose a label initialization strategy, in which GLLPA assigns the nodes locating in the same position with the same label in community detection. In order to draw networks to uniform layout, the attractive forces among nodes should close to the repulsive forces [35], and their total stress condition also should form equilibrium condition.

Table 1
Notations and their descriptions.

Notation	Description
$G(V, E)$	G: network, V: node set, E: edge set
v_i	The ith node in V
e_i	The ith edge in E
n	The number of nodes
m	The number of edges
k_{v_i}	The degree of node v_i
$N(v_i)$	The neighbor nodes of node v_i
$M(v_i)$	The label set of node v_i
$(l_j^{v_i}, c_j^{v_i})$	The jth label with belonging coefficient of node v_i
H	The number of labels in $M(v_i)$
p	The layout of G
S	Layout space
$p(v_i) = (x, y)$	The position of node v_i in S with abscissa x and ordinate y
$F^a(v_i, v_j)$	The node attraction between node v_i and v_j
$F^r(v_i, v_j)$	The node repellent between node v_i and v_j
$F(v_i)$	The resultant force acting on node v_i
a	Attractive parameter
r	Repulsive parameter
$\varepsilon(p\|G)$	The energy of layout p for network G
I_{v_i}	The node influence of node v_i
C_{v_i}	The centrality of node v_i
$LP_{l,v_j\to v_i}$	The label influence of label l when it is launched from node v_j to node v_i
$LP_j^{v_i}$	The label influence of the jth label of node v_i
$D = \{D_1, \ldots, D_i, \ldots, D_d\}$	D: initial position set after compact layout, D_i: the ith position, d: the number of positions
CS	Community set
α	Influence factor
β	Filtering threshold
γ	Step size used for minimizing energy

3.4. Node influence

Node influence reflects the weight of nodes in networks. Inspired from the roles of ants (i.e., the influence of ants comes from the influence of ant itself and its followers), we define node influence as the weighted average of personal influence and neighboring influence, which can be written as

$$I_{v_i} = \alpha C_{v_i} k_{v_i} + (1-\alpha) \sum_{v_j \in N(v_i)} \frac{k_{v_j}}{\sum_{v_k \in N(v_i)} k_{v_k}} * C_{v_j} k_{v_j}, \tag{4}$$

where $C_{v_i} k_{v_i}$ is the personal influence of node v_i, as denoted by centrality C_{v_i} and degree k_{v_i} of node v_i in networks. The second item denotes the neighbor influence of node v_i, which is affected by three factors, the weight, centrality and degree of neighbor v_j. Influence factor $\alpha \in [0, 1]$ is used to balance the personal and neighbor influence of node v_i. Centrality C_{v_i} of node v_i measures its global centrality according to the distances among nodes in networks, which is defined as

$$C_{v_i} = \frac{N-1}{\sum_{v_j \in N(v_i)} \|p(v_i) - p(v_j)\|} = \frac{N-1}{\sum_{v_j \in N(v_i)} \sqrt[a]{\frac{F^a(v_i, v_j)}{\overrightarrow{p(v_i)p(v_j)}}}}. \tag{5}$$

With the change of layout, node centrality varies, then node influence can change dynamically to mimic the improvement of animal roles. In order to deal with the randomness problem, we update the labels of nodes in one iteration in the ascending order of node influence because the nodes with small node influence can be affected by other nodes easily.

3.5. Label influence

Inspired from the behaviors of bats, we design label influence by node influence, node attraction and the belonging coefficients of labels. Label influence reflects the weight of the label when the label arrives at another node, which is defined as

$$LP_{l,v_j\to v_i} = \frac{I_{v_j} \times c^{v_j}}{\|p(v_i) - p(v_j)\|} = \frac{I_{v_j} \times c^{v_j}}{\sqrt[a]{\frac{F^a(v_i, v_j)}{\overrightarrow{p(v_i)p(v_j)}}}}. \tag{6}$$

Here, c^{v_j} is the belonging coefficient of label l of node v_j. The labels of a node will update to more influential labels.

4. The proposed algorithm

4.1. Algorithm description

GLLPA consists of two parts: network layout and community detection, which are elaborated in Sections 4.1.1 and 4.1.2, respectively. The network layout part of GLLPA first draws networks to compact layout, then it draws networks to uniform layout. When GLLPA finishes the compact layout of networks, it starts community detection part using the layout information. The pseudo code of GLLPA is shown in **Algorithm 1**.

Layout Part

In layout part, each node is mapped to a position in a low-dimensional space. In this paper, we consider the 2-dimensional Euclidean space. The layout is drawn according to the force-directed method with (a, r)-energy model. Referencing [33,35], we use (a, r)-energy model and set parameters $a = -0.95$ and $r = -1$ to increase node attraction to draw networks to compact layout, as shown in Fig. 1(a), in which nodes tend to gather together under large node attraction, whose details are presented in Appendix. Then we set parameters $a = 2$ and $r = -1$ to draw networks to uniform layout, as shown in Fig. 1(b), in which nodes are distributed uniformly in layout area. Here, $\|p(v_i) - p(v_j)\|^{(-1+1)}/(-1+1)$ is read as $\ln \|p(v_i) - p(v_j)\|$ when $r = -1$ because x^{-1} is the derivative of $\ln x$ [33]. The process of compact layout and uniform layout is the same, as shown in Fig. 4(a).

Detection Part

When the networks are drawn to compact layout, detection part starts based on the current layout. Nodes locate the same position in the compact layout belong to the same community possibly, so we give them the same label in the initialization of community detection. Then a node and label influence based label propagation process is proposed to discover communities by using the attractive forces in the uniform layout. The process of detection part is shown in Fig. 4(b). In order to use the attractive force to compute node influence and label influence, we combine uniform layout and label propagation, as shown in **Algorithm 1**.

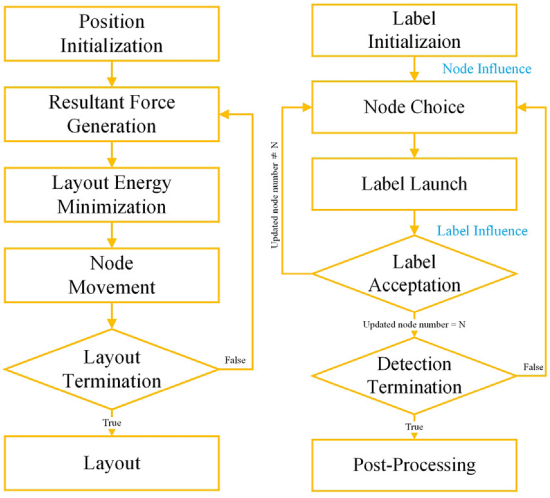


Fig. 4. The processes of layout and detection parts of GLLPA.

Algorithm 1 GLLPA

Input: $G = (V, E)$, a, r, α, β;
Output: Community set CS;

Set parameters $a = -0.95, r = -1$; //Conduct compact layout
$p =$ Position Initialization (G);
repeat
  $t_1 = 0$;
  $F =$ Resultant Force Generation (a, r);
  $\gamma =$ Layout Energy Minimization (F, a, r);
  $p =$ Node Movement (γ);
  $t_1 = t_1 + 1$;
until Layout Termination () == True or $t_1 = MaxLayoutTimes$;
Set parameters $\alpha = 0.4, \beta = 2$; //Start community detection
Label Initialization (p);
Set parameters $a = 2, r = -1$; //Conduct uniform layout
repeat
  $t_2 = 0$;
  $F =$ Resultant Force Generation (a, r);
  $\gamma =$ Layout Energy Minimization (F, a, r);
  Node Choice (α);
  repeat
    Label Launch ();
    Label Acceptation (β);
  until all nodes in V are updated;
  $p =$ Node Movement (γ);
  $t_2 = t_2 + 1$;
until Detection Termination () == True or $t_2 = MaxDetectionTimes$;

$CS =$ Post-processing ();
return CS;

4.1.1. Network layout

The procedure of network layout includes position initialization, resultant force generation, layout energy minimization, node movement and layout termination, as shown in Fig. 4(a). Whether networks are drawn to compact layout or uniform layout, the procedure is the same, only parameters a and r are different [33,39].

Step 1: Position Initialization

Each node v_i in V is mapped to random position $p(v_i)$ in a 2-dimensional layout space, such as

$$p(v_i) = (x \in [-\frac{W}{2}, \frac{W}{2}], y \in [-\frac{L}{2}, \frac{L}{2}]). \tag{7}$$

Step 2: Resultant Force Generation

The attractive and repulsive forces acting on node v_i are calculated. Then the resultant force is derived as

$$F(v_i) = \sum_{v_j \in N(v_i)} F^a(v_i, v_j) + F^r(v_i, v_j). \tag{8}$$

Step 3: Layout Energy Minimization

Since (a, r)-energy model is general enough to represent many force-directed graph layout models [39], we set parameters $a = -0.95$ and $r = -1$ to increase the attractive forces among nodes to draw networks for gathering nodes as compact layout [33]. Then we set parameters $a = 2$ and $r = -1$ to balance the attractive and repulsive forces among nodes to draw networks for distributing nodes evenly as uniform layout [35].

Thus, for compact layout, Eq. (3) can be re-written as

$$\begin{aligned} \varepsilon(p|G) = & \sum_{e_{v_i v_j} \in E} \left\| p(v_i) - p(v_j) \right\|^{0.05} * 20 \\ & - \sum_{e_{v_i v_j} \in E} k_{vi} k_{vj} \ln \left\| p(v_i) - p(v_j) \right\|. \end{aligned} \tag{9}$$

For uniform layout, Eq. (3) can be re-written as

$$\begin{aligned} \varepsilon(p|G) = & \sum_{e_{v_i v_j} \in E} \left\| p(v_i) - p(v_j) \right\|^{3} * (1/3) \\ & - \sum_{e_{v_i v_j} \in E} k_{vi} k_{vj} \ln \left\| p(v_i) - p(v_j) \right\|. \end{aligned} \tag{10}$$

Energy is a quantitative measurement of network layout. In general, smaller energy indicates better layout [33], then we want to gain the layout with the smallest energy. When nodes move under forces, the layout and its energy will change. Thus, we calculate the layout energy under different step size γ with the resultant force by Eq. (11) to gain the minimum energy in current iteration [33].

$$\gamma = \arg \min_{\gamma \in \{2^{-i} | i = 0, \ldots, 10\}} \varepsilon(p + \gamma F|G) \tag{11}$$

Here, F represents the resultant forces of all nodes in networks and γ varies from 0 to 10 with interval 1.

Step 4: Node Movement

If the layout energy can be decreased under certain step size γ with the resultant force, then each node v_i in V obtains new $p(v_i)$ by

$$p(v_i) = p(v_i) + \gamma F(v_i). \tag{12}$$

Step 5: Layout Termination

If the layout energy cannot be reduced, then the layout reaches equilibrium state.

4.1.2. Community detection

The procedure of community detection contains label initialization, node choice, label launch, label acceptation, detection termination and post processing, as shown in Fig. 4(b).

Step 1: Label Initialization

Based on the compact layout, if there are some nodes locating in the same position, then they are assigned with the same label in this step.

Set $V' = V$ and $M(v_i) = \{(l_1^{v_i} = j, c_1^{v_i} = 1)\}$ for each node v_i in D_j. Here, V' denotes the node set, in which nodes have not been updated, and $D = \{D_1, \ldots, D_j, \ldots, D_d\}$ where the current layout has d positions and multiple nodes may be on one of these positions.

Step 2: Node Choice

In this step, node v_i is chosen to update its labels.

(1) Node influence I_{v_i} of node $v_i \in V$ is computed, then the nodes in V' are ordered in the ascending order of node influence.
(2) Node v_i satisfying $I_{v_i} = \min_{v_j \in V'}(I_{v_j})$ is selected, then set $M(v_i) = \emptyset$.

Step 3: Label Launch

The neighboring nodes of node v_i launch their labels to node v_i.

(1) For each node v_j in $N(v_i)$, label l^{v_j} satisfying $c^{v_j} = \max_{(l^{v_x}, c^{v_x}) \in M(v_j)}(c^{v_x})$ is selected, then node v_j launches label l^{v_j} to node v_i. Here, GLLPA sends the label with the largest belonging coefficient for handling the instability problem.
(2) When label l^{v_j} arrives at node v_i, the label is assigned label influence $LP_{l^{v_j}, v_j \to v_i}$, then $M(v_i) = M(v_i) \cup \{(l^{v_j}, LP_{l^{v_j}, v_j \to v_i})\}$.
(3) Nodes receive the labels from different neighbor nodes, then they may be same. Thus, we sum the label influence of the labels with the same id.

Step 4: Label Acceptation

This step is used to accept important labels and filter the labels with small belonging coefficients. Thus, referencing [27], we use threshold $\beta \in [1, +\infty]$ to filter the labels with small belonging coefficients and control the number of communities that nodes can belong to.

(1) By normalizing the label influence of labels in $M(v_i)$, we obtain $M(v_i) = \{(l_1^{v_i}, c_1^{v_i}), \ldots, (l_j^{v_i}, c_j^{v_i}), \ldots, (l_H^{v_i}, c_H^{v_i})\}$, where $c_j^{v_i} = \frac{LP_{l_j^{v_i}}}{\sum_{k=1}^{H} LP_{l_k^{v_i}}}$. Here, $LP_{l_j^{v_i}}$ denotes the label influence of the jth label of node v_i.
(2) For each $(l_j^{v_i}, c_j^{v_i})$ in $M(v_i)$, if $c_j^{v_i} < \frac{1}{\beta}$, then $M(v_i) = M(v_i) - (l_j^{v_i}, c_j^{v_i})$.
(3) By normalizing the belonging coefficients of labels in $M(v_i)$, GLLPA accepts rest labels and obtain updated $M(v_i)$ of node v_i, then it sets $S = S - \{v_i\}$.

Step 5: Detection Termination

The stop criterion is defined as same as COPRA [27] in GLLPA. The proposed algorithm calculates minimal number set m_t of nodes signed by each community identifier. If $m_t = m_{t-1}$ or the metric cannot be increased, GLLPA conducts post-processing, else sets $V' = V$ and returns to the step of node choice for next iteration.

Step 6: Post-Processing

Each node whose labels contain community identifier l is allocated to community l. Overlapping nodes are allocated to multiple communities.

4.2. Complexity analysis

The time complexity of GLLPA is estimated as follows.

Layout part

(1) Position initialization: Each node is mapped to a position in layout area with time $O(n)$.
(2) Resultant force generation: Computing the attractive, repulsive and resultant forces takes time $O(m)$.
(3) Layout energy minimization: The time of gaining the minimum of energy is $O(n)$.
(4) Node movement: Updating the positions of all nodes takes time $O(n)$.
(5) Layout termination: The time of judging whether layout part stops can be negligible.

Detection part

(1) Label initialization: Each node is given a label takes time $O(n)$.
(2) Node choice: Calculating the node influence of all nodes needs time $O(n)$ and sorting nodes according to node influence needs time $O(n \log n)$.
(3) Label launch: Sorting the labels of each node by their belonging coefficient needs the worst time $O(|N(v_i)| \times n_1 \log n_1)$, where n_1 is the maximum number of labels of each node, $n_1 \ll n$. Thus, this step takes constant time.
(4) Label acceptation: Filtering the labels of node v_i through threshold β takes the worst time $O(n_2)$, where n_2 is the maximum number of labels arriving at node v_i, $n_2 \ll n$. Therefore, this step takes constant time.
(5) Detection termination: As same as COPRA [27], it takes time $O(\beta n)$.
(6) Post-processing: As same as COPRA [27], it takes time $O(\beta^3 n + \beta(n + m))$.

Thus, the total time complexity of GLLPA is $O((2t_1 + 4t_2 + \beta^3 + 2\beta + 1)n + t_2 n \log n + (t_1 + t_2 + \beta)m)$, where t_1 is the iteration times of compact layout and t_2 is the iteration times of community detection. In conclusion, as $t_1, t_2, \beta \ll n$, the total time complexity of GLLPA is near $O(n \log n + c_1 m)$, where c_1 is a constant.

The space complexity of GLLPA is estimated as follows. We mainly analyze the space requirement of GLLPA. First, nodes and edges need space $O(n)$ and $O(m)$, respectively. Second, each node most needs $2maxk$ space to store labels and their belonging coefficients, then all nodes need space $O(2maxkn)$. Third, the positions and displacements of nodes take $O(2n)$ space. Finally, node attraction, node repellent and node influence needs space $O(n)$, respectively. Label influence can share the space with belonging coefficient. Thus, the total space complexity of GLLPA is about $O((6 + 2maxk)n + m)$. For $maxk \ll n$, then it is near $O(n + m)$.

5. Experiments

In this section, extensive experiments are conducted with all algorithms on the hardware of Intel Core i3-4370 3.80 GHz processor and 8 GB memory. Each experiment includes 50 independent trials. The synthetic and real-world networks are introduced firstly. Then the baselines and evaluation criteria are described, and we analyze the influence of parameters α and β on the results of GLLPA. Finally, we analyze the experimental results.

5.1. Datasets

In this paper, 16 synthetic and 15 real-world networks are adopted to analyze the effectiveness of GLLPA. Synthetic networks are generated by LFR benchmark [52], which introduces heterogeneity into the degree and community size distributions of networks and is closer to the features observed in real-world networks. Above two distributions are governed by power laws with exponents τ_1 and τ_2, respectively. LFR also can control the topology of networks by adjusting following parameters: the number of nodes n, average degree $\langle k \rangle$, maximum degree $maxk$, the minimum and maximum of community sizes $minc$ and $maxc$, mixing parameter mu, the number of overlapping nodes O_n and the number of the memberships of overlapping nodes O_m. Parameter mu denotes the expected fraction of the edges of a node connecting to other communities. The number of overlapping nodes O_n is specified and each node is assigned to $O_m > 1$ communities for generating the networks with overlapping communities (If $O_m = 1$, then the networks are generated without overlapping communities). The networks with smaller mu or O_m have clearer community structures, then the algorithms can detect them more easily [52].

Table 2
Description of synthetic networks.

Network	n	$\langle k \rangle$	$maxk$	$minc$	$maxc$	mu	O_n	O_m
LFR-1	5000	10	50	20	100	0.1	0	1
LFR-2	5000	10	50	20	100	0.3	0	1
LFR-3–9	5000	10	50	20	100	0.1	500	2–8
LFR-10–16	5000	10	50	20	100	0.3	500	2–8

Table 3
Description of real-world networks.

Network	Reference	n	m	c	$\langle k \rangle$	dia	cc
Karate	[53]	34	78	2	4.588	5	0.588
Dolphins	[54]	62	159	2	5.129	8	0.303
Polbook	[55]	105	441	2	8.400	7	0.488
Football	[56]	115	615	12	10.661	4	0.403
Netscience	[57]	1589	2742	16	3.451	17	0.878
Lesmis	[58]	77	254	–	6.579	5	0.736
Jazz	[59]	198	2742	–	27.697	6	0.633
Email	[60]	1133	5451	–	9.624	8	0.255
Power	[61]	4941	6594	–	2.669	46	0.107
PGP	[62]	10680	24316	–	4.554	24	0.440
Cond2003	[63]	31163	120029	–	7.703	16	0.723
Enron	[64]	36692	183831	–	10.020	13	0.716
Cond2005	[63]	40421	175693	–	8.693	18	0.719
DBLP	[65]	317080	1049866	–	4.929	21	0.632
Amazon	[65]	334863	925872	–	5.530	44	0.396

"c" denotes the number of communities, "dia" denotes the diameter of networks and "cc" denotes the average clustering coefficient of networks.

We use the LFR networks with node degree and community sizes governed by power law distributions with exponents $\tau_1 = 2$ and $\tau_2 = 1$, respectively. The number of nodes $n = 5000$. The average degree $\langle k \rangle = 10$, which is the same order as most real-world social networks. Maximum degree $maxk = 50$, the minimum and maximum of community sizes $minc = 20$ and $maxc = 100$ and mixing parameter mu equals to 0.1 and 0.3. The overlapping degree is determined by parameters O_n and O_m. We make the former to be 10% of n and the latter varies from 2 to 8 indicating the diversity of overlapping nodes when mu keeps unchanged. We also construct two networks with $O_n = 0$ and $O_m = 1$ when $mu = 0.1$ and 0.3 to form the networks with non-overlapping communities. The LFR benchmarks used in the experiments are given in Table 2. The real-world networks are divided into two categories with true community structures known or not listed in Table 3.

5.2. Baseline algorithms

To evaluate the effectiveness of GLLPA, we compare it with state-of-the-art algorithms listed below.

- COPRA [27] is a label propagation based model for discovering communities by assigning multiple labels with belonging coefficients to a node.
- SLPA [28] is a label propagation based approach to detect communities by allocating nodes with memory spaces to store received labels.
- DLPA$^+$ [30] considers the weight of nodes' neighbors and introduces confidence variance to improve stability to discover communities.
- WLPA [31] is also a label propagation based algorithm for detecting communities by considering the weight of nodes and labels.
- LinkLPA [25] is an edge-based label propagation algorithm, which transforms node partition problem into edge partition problem to discover communities.
- LPA_NI [32] is a node and label weight based label propagation method to explore the weight of nodes and labels based on Bayesian network for discovering communities.
- NGLPA [21] is proposed to detect non-overlapping communities based on node gravitation and node similarity.
- MWLP [23] is a motif-based label propagation algorithm for detecting communities by unifying both the higher-order structure and the original lower-order structure.
- GLPA [26] is a graph based label propagation algorithm to discover communities incorporating the node similarity and connectivity information.
- ComE [18] is an embedding-based community detection method that jointly solves community embedding, community detection and node embedding together.
- DANMF [17] is a NMF-based community detection method that uses a deep autoencoder-like architecture.

Here, we present the time and space complexities of GLLPA and baselines, as shown in Table 4. For time complexity, we can find that the time complexity of label propagation based algorithms can be mainly classified as three types, (1) the time complexity is $O(c_1 m + c_2 n)$ if the methods do not sort nodes, (2) the time complexity is $O(n \log n + c_3 m + c_4 n) \approx O(n \log n + c_3 m)$ if the methods sore nodes in prescribed rules and (3) the time complexity is $O(n^2)$ if the methods compute node similarity, where $c_1, \ldots, c_5$ are different coefficients for different methods. For space complexity, we can find that it can be classified as two types, (1) the space complexity is $O(c_6 n + c_7 m)$ if the methods need to store nodes and edges with other the same scale information, (2) the space complexity is $O(n^2 + c_8 n + c_9 m) \approx O(n^2 + c_9 m)$ if the methods needs to store node similarity or full adjacency matrix, where $c_6, \ldots, c_9$ are different coefficients for different methods.

5.3. Evaluation criteria

To compare different algorithms with respect to performance and focus on the results of community detection, two criteria [66], normalized mutual information (NMI) [67] and overlap modularity (Q_{OV}) [8], are used to evaluate the quality of communities. Also, we compute the running time of methods to evaluate their execution efficiency.

Table 4
The complexity of methods.

Methods	Time complexity	Space complexity
COPRA	$O((\gamma^3+\gamma)n+t\gamma m\log(\gamma m/n)+\gamma m)$	$O((1+2maxk)n+m)$
SLPA	$O(n+tm)$	$O((1+t)n+m)$
WLPA	$O((\gamma^3+\gamma)n+t\gamma m\log(\gamma m/n)+\gamma m)$	$O((1+2maxk)n+m)$
DLPA$^+$	$O(n\log n+n+tm)$	$O((1+3maxk)n+m)$
LinkLPA	$O(cn+m)$	$O(n+2m)$
LPA_NI	$O(n\log n+3n+2tm)$	$O((2+2maxk)n+m)$
NGLPA	$O((n+1)logn+(2+maxk)n+tm)$	$O(n^2+4n+m)$
MWLP	$O(m^{1.5})$	$O(2n+2m)$
GLPA	$O(n^2)$	$O(n^2+n+m)$
ComE	$O(\gamma\ln+n+t_1((t_2+1)nc+\gamma\ln+c+m))$	$O((1+d)n+m)$
DANMF	$O(p(t_p+t_f)(n^2r+nr^2))$	$O(n^2+n+m)$
GLLPA	$O(n\log n+c_1m)$	$O((6+2maxk)n+m)$

For LinkLPA, c is the number of clusters before merging.
For ComE, γ is the number of nodes in a path, l is walk length, c is the number of communities, t_1 is the iteration time and t_2 the sub-iteration time.
For DANMF, p is the number of layers, t_p is the number of iterations to achieve convergence, r is the maximal layer size out of all layers and t_f is the number of iterations in the fine-tuning process.

Table 5
LFR-0 for parameter analysis.

Network	n	$\langle k\rangle$	$maxk$	$minc$	$maxc$	mu	O_n	O_m
LFR-0	5000	10	50	20	100	0.5	500	6

Normalized mutual information (*NMI*) [67] varies between 0 and 1 and is used to show the difference between the divisions obtained by community detection algorithms and true community structures. The larger the *NMI*, the smaller the difference. Overlap modularity (Q_{OV}), the extension of Newman's modularity [8], varies between 0 and 1 and is used to reflect the quality of the divisions assessed by the relative density of the edges within communities and between communities. As Q_{OV} function, we adopt the one used in [27], $f(x)=60x-30$. The larger the Q_{OV}, the better the quality of communities. If the true communities of networks are known, *NMI* and Q_{OV} are both adopted, otherwise only Q_{OV} is adopted.

5.4. Parameter discussion

Here, we investigate the impact of parameters α and β on the results of GLLPA, in which α controls the effect of personal and neighbors' influence for node influence and β is the filtering threshold to control the number of communities that nodes can belong to. We choose LFR-0 shown in Table 5 for parameter analysis, which has biggish *mu* and O_m, so the algorithms cannot detect communities easily because communities become fuzzy when the networks have large *mu* and O_m. If GLLPA with some parameters can obtain satisfying results in LFR-0, then we infer that the desirable partitions can be obtained when GLLPA is applied to the networks with small *mu* and O_m. Meanwhile, we choose a real-world network, i.e., Football network, to conduct parameter analysis for analyzing the impact of parameters α and β on GLLPA in real-world networks. Parameter α varies from 0 to 1 with interval 0.1 and parameter β varies from 1 to 9 with interval 1. The effects of α and β in terms of *NMI* and Q_{OV} are presented in Fig. 5.

For LFR-0, we can find that parameter α has little impact on the results of *NMI* and Q_{OV}, while parameter β has large influence on these results, as shown in Figs. 5(a) and 5(b). GLLPA with $\alpha=0.4$ outputs the highest results in terms of *NMI* and GLLPA with $\alpha=0.3$ achieves the best results in terms of Q_{OV} when β is fixed to 1 or 2. It can be seen that the neighbor influence is crucial for node influence. As a trade-off, we choose $\alpha=0.4$ to detect the results which are closer to true communities. The results of *NMI* and Q_{OV} decline with the increase of β. If nodes belong to more than three communities, then the results become poor. Maybe it is because overlapping nodes are not closed in each community and the methods cannot detect accurate overlapping nodes in multiple communities easily. Analyzing these results, we can consider that GLLPA obtains better *NMI* and Q_{OV} with $\beta=1$ and 2 than other values of β, then we choose $\beta=2$ for discovering overlapping community.

For Football network, as shown in Figs. 5(c) and 5(d), parameter α also has small impact on the results of *NMI* and Q_{OV}, while parameter β has large influence on these results, which are similar with the conclusion in LFR-0. The results for Football network with changed α values shows larger fluctuation than those for LFR-0, which reflects that the results are more sensitive to parameter α for real-world networks. However, the difference among the results for Football network with different values of α is small when $\beta=1$ and 2. Then we can fine α to obtain the optimum in real-world networks. For consistency, we also choose $\alpha=0.4$. Similar with the results in LFR-0, GLLPA obtains better *NMI* and Q_{OV} with $\beta=1$ and 2 than other values of β for Football network, then we also choose $\beta=2$ for discovering overlapping community.

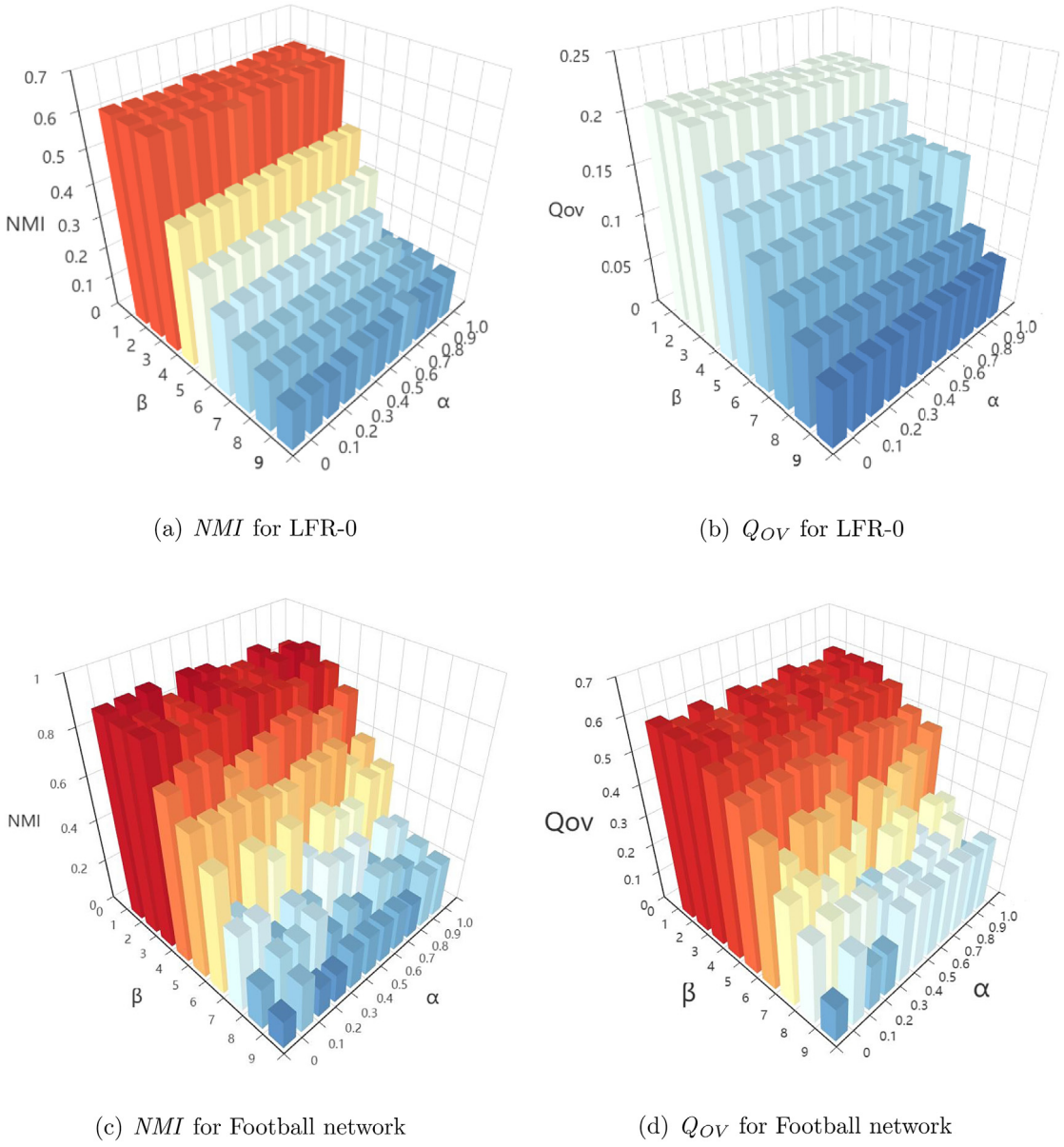
By discussing parameters α and β, we infer that GLLPA can achieve better performance when $\alpha=0.4$ and $\beta=2$. Therefore, we choose GLLPA with them to compare with state-of-the-art algorithms in networks.

5.5. Results for synthetic networks

5.5.1. Results for the synthetic networks with non-overlapping communities

The average values and standard deviation of *NMI* and Q_{OV} and the average value of running time for the synthetic networks with disjoint communities are shown in Table 6. The number in bracket is the rank of methods for each network under different criteria and their average rank is given at the bottom. The smaller the average rank, the better the performance of methods. As shown in Table 6, we find that the majority results (NMI_{avg}, Q_{OVavg} and running time) in LFR-1 are better than the results in LFR-2 obtained of each algorithm, since LFR-1 has smaller *mu* than LFR-2 so that the communities in LFR-1 can be detected easily.

For average rank, we can find that the results can be mainly classified as four types. (1) [weak results, high efficiency] COPRA, DLPA$^+$, LinkLPA, MWLP and GLPA output communities much sooner than some methods (e.g., SLPA, LPA_NI and NGLPA), but their NMI_{avg} and Q_{OVavg} are poorer than them. (2) [weak results, low efficiency] the NMI_{avg} and Q_{OVavg} of ComE and DANMF are weaker than most methods and their running time is also



(a) *NMI* for LFR-0

(b) Q_{OV} for LFR-0

(c) *NMI* for Football network

(d) Q_{OV} for Football network

Fig. 5. Results of parameter analysis.

higher than other methods. (3) [good results, low efficiency] SLPA, LPA_NI and NGLPA obtain good results, however, they cost high running time. (4) [good results, high efficiency] WLPA ranks the second in terms of NMI_{avg} with less time, especially, WLPA obtains the best NMI_{avg} in LFR-2, however, its Q_{OVavg} is the fifth among all experimental methods. For our proposed method, GLLPA achieves the highest NMI_{avg} and suboptimal Q_{OVavg} sooner than experimental methods. Although GLLPA draws networks costing some time, the operation can reduce the iterations of community detection by discovering accurate nodes which should be in the same community according to the label initialization strategy and node attraction. Label influence can filter unimportant labels to speed up iterating. In addition, NGLPA obtains the best Q_{OVavg} in the synthetic networks with non-overlapping communities and is better than GLLPA, which may because NGLPA chooses the label with maximum gravitation to update the current label of nodes for increasing the closeness among community nodes. Meanwhile, most methods are designed for discovering overlapping communities, then nodes may be assigned to multiple communities, then the closeness decreases in these networks with non-overlapping communities, however, NGLPA is designed for discovering disjoint communities, then it is suitable for these networks and can obtain communities with high internal tightness.

Considering the stability of these algorithms according to the average rank, it can be mainly classified as three types. (1) [relatively weak stability in terms of *NMI* and Q_{OV}] The stability of COPRA, LinkLPA, MWLP, ComE and DANMF is weak for both *NMI* and Q_{OV}. Among them, COPRA and MWLP choose nodes randomly to update their labels. LinkLPA chooses edges randomly to update their labels. ComE samples nodes and their neighbors to train node embeddings and lead to inferior stability. DANMF randomly initializes multiple factor matrices $\boldsymbol{U_1}, \ldots, \boldsymbol{U_p}$ and $\boldsymbol{V_1}, \ldots, \boldsymbol{V_p}$ to conduct multilevel matrix factorization, which results in poor stability. (2) [relatively good stability in terms of *NMI* or Q_{OV}] SLPA, DLPA$^+$, WLPA and NGLPA have better stability in one of *NMI* and Q_{OV} than some methods in LFR-1 or LFR-2. SLPA stores the received labels of nodes and chooses labels with large occurrence frequency as community labels of nodes that can reduce randomness. DLPA$^+$, WLPA and NGLPA update nodes'

Table 6
Results for the synthetic networks with non-overlapping communities.

Criterion	Network	COPRA	SLPA	DLPA^{+}	WLPA	LinkLPA	LPA_NI	NGLPA	MWLP	GLPA	ComE	DANMF	GLLPA
NMI_{avg}	LFR-1	0.9853 (10)	0.9994 (2)	0.9887 (9)	0.9980 (4)	0.9924 (7)	0.9987 (3)	0.9973 (6)	0.8676 (12)	0.9974 (5)	0.9823 (11)	0.9901 (8)	**0.9997 (1)**
	LFR-2	0.9859 (5)	0.9931 (3)	0.9414 (9)	**0.9979 (1)**	0.9682 (7)	0.9847 (6)	0.9916 (4)	0.7348 (12)	0.9369 (10)	0.9517 (8)	0.7415 (11)	0.9969 (2)
	Average rank	7.5	2.5	9.0	2.5	7.0	4.5	5.0	12.0	7.5	9.5	9.5	**1.5**
NMI_{std}	LFR-1	0.0046 (10)	0.0005 (2)	0.0013 (7)	0.0011 (6)	0.0047 (12)	0.0008 (5)	0.0013 (7)	0.0007 (4)	**0.0002 (1)**	0.0046 (10)	0.0028 (9)	0.0005 (2)
	LFR-2	0.0041 (7)	0.0035 (6)	0.0015 (5)	0.0011 (2)	0.0046 (8)	0.0012 (4)	0.0047 (9)	0.0113 (10)	0.0011 (2)	0.0114 (11)	0.0243 (12)	**0.0010 (1)**
	Average rank	8.5	4.0	6.0	4.0	10.0	4.5	8.0	7.0	**1.5**	10.5	10.5	**1.5**
Q_{OVavg}	LFR-1	0.4259 (10)	0.4467 (4)	0.4423 (8)	0.4443 (7)	0.4239 (11)	0.4467 (4)	**0.4473 (1)**	0.4025 (12)	0.4470 (3)	0.4369 (9)	0.4451 (6)	0.4471 (2)
	LFR-2	0.3353 (7)	0.3437 (3)	0.3381 (5)	0.3366 (6)	0.3239 (9)	0.3437 (3)	**0.3468 (1)**	0.3124 (11)	0.3195 (10)	0.3264 (8)	0.2998 (12)	0.3459 (2)
	Average rank	8.5	3.5	6.5	6.5	10.0	3.5	**1.0**	11.5	6.5	8.5	9.0	2.0
Q_{OVstd}	LFR-1	0.0078 (12)	0.0012 (5)	0.0016 (7)	0.0017 (8)	0.0074 (11)	0.0002 (1)	0.0002 (1)	0.0031 (9)	0.0003 (4)	0.0037 (10)	0.0014 (6)	**0.0002 (1)**
	LFR-2	0.0028 (9)	0.0019 (8)	0.0007 (3)	0.0014 (7)	0.0012 (6)	0.0011 (5)	0.0008 (4)	0.0056 (11)	0.0006 (2)	0.0053 (10)	0.0066 (12)	**0.0002 (1)**
	Average rank	10.5	6.5	5.0	7.5	8.5	3.0	2.5	10.0	3.0	10.0	9.0	**1.0**
Time (s)	LFR-1	130.1102 (6)	847.8163 (10)	462.4332 (8)	48.1446 (2)	78.6285 (4)	494.3510 (9)	346.1910 (7)	79.7413 (5)	56.6079 (3)	1478.2553 (12)	895.1611 (11)	**47.9534 (1)**
	LFR-2	189.9801 (7)	941.3162 (10)	77.2122 (5)	65.5102 (2)	73.4443 (4)	480.5123 (9)	383.0065 (8)	73.3862 (3)	90.8754 (6)	1554.4403 (12)	996.0928 (11)	**65.2008 (1)**
	Average rank	6.5	10.0	6.5	2.0	4.0	9.0	7.5	4.0	4.5	12.0	11.0	**1.0**

labels according to the designed order to improve instability, then they should obtain good stability. However, they are more inferior than some methods, then they obtains relatively weak stability. (3) [relatively good stability in terms of *NMI* and Q_{OV}] GLPA and GLLPA have good stability in terms of both *NMI* and Q_{OV}. For GLPA, it obtains tied for the first with GLLPA in terms of *NMI*, specially, it ranks the first in LFR-1. GLPA constructs super nodes to directly manipulate many nodes once to discover communities, which reduces randomness. For GLLPA, it outputs more steady results than other methods, since GLLPA determines the node order of label updating according to node influence, the order of label launching according to belonging coefficient and the order of label accepting according to label influence. However, we find GLLPA still unstable. The compact layout result of a network may be not always the same, then the label initialization of community detection is different. Meanwhile, when the number of the most influential label of a node is more than one, GLLPA randomly sends one of the labels to other nodes, which causes the instability problem. In general, GLLPA outperforms most methods for the synthetic networks with non-overlapping communities.

5.5.2. Results for the synthetic networks with overlapping communities

The results of *NMI*, Q_{OV} and running time for the synthetic networks with overlapping communities are shown in Fig. 6. Meanwhile, the average rank of methods is shown in Table 7. As shown in Fig. 6(a), (b), (e) and (f), the results of NMI_{avg} and Q_{OVavg} decline with the increase of O_m and their values when $mu = 0.3$ are smaller than those when $mu = 0.1$, which also illustrates that communities become more intricate and can be discovered more hardly with the increase of O_m and *mu*.

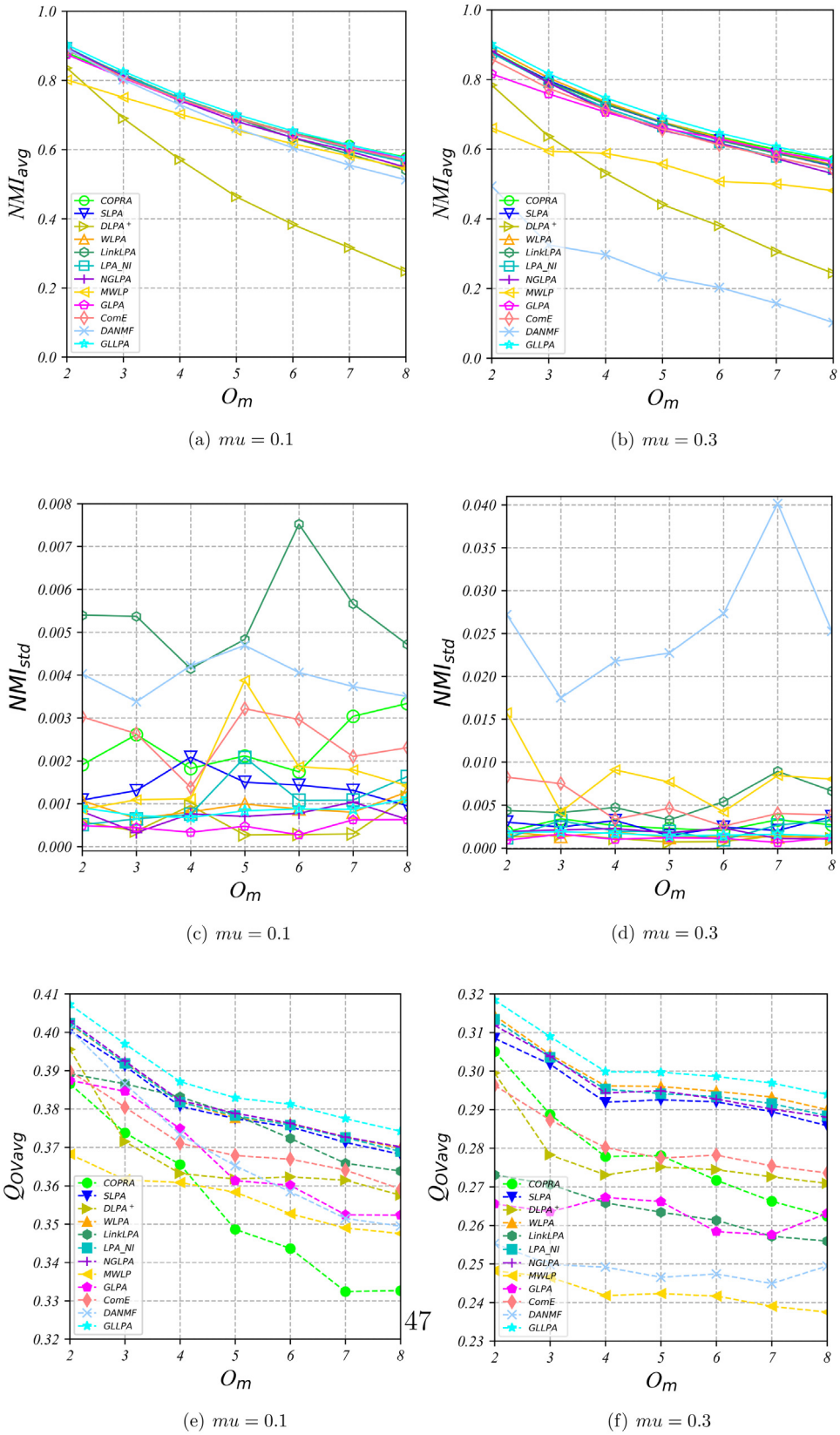
In terms of NMI_{avg}, as shown in Fig. 6(a), (b) and Table 7, all methods except DLPA$^+$, MWLP and DANMF behave similarly. We can find that GLLPA outperforms other methods and the eight algorithms (COPRA, SLPA, WLPA, LinkLPA, LPA_NI, NGLPA, GLPA and ComE) obtain similar and suboptimal results. Among them, COPRA assigns nodes with multiple labels to learn the community membership of nodes in networks. SLPA analyzes the accepted labels of a node to decide its community assignment. WLPA considers the label weight of a node by the ratio of its degree to the average degree of networks to reflect its ownership. LinkLPA detects edge communities and transforms them to node communities to detect node partitions. In LPA_NI, the weight of nodes and labels is defined by the priori importance and degree of nodes based on a Bayesian model to recognize the membership of nodes. NGLPA computes node weight and node similarity for measuring node gravitation to differentiate the nodes in different communities. Although NGLPA cannot find overlapping communities, it can find more tight groups by node gravitation. GLPA constructs label propagation graph and merges super nodes to find communities with similar nodes. ComE learns the community embeddings with node embeddings to improve the quality of community results. For our proposed method, GLLPA utilizes node attraction to catch the interactions among nodes and makes use of dynamic node and label influence for capturing the importance change of nodes and labels to distinguish nodes in different communities. In summary, they obtain good NMI_{avg}. Specially, GLLPA achieves the best NMI_{avg}. On the other hand, MWLP and DANMF are slightly worse than above methods, since they may be slightly sensitive to the increase of network complexity with the variation of O_m. The results of DLPA$^+$ are poorer and falls faster than those of other methods, since the neighbor confidence in DLPA$^+$ may not reflect the community membership of nodes accurately in large networks.

Considering the standard deviation of *NMI* (NMI_{std}), as shown in Fig. 6(c) and (d) and Table 7, we can find that methods' stability is similar with the results in synthetic networks with non-overlapping communities for the same reason mentioned in Section 5.5.1. The stability of LinkLPA and DANMF is poorer than other methods and shows large fluctuation, which is the worst in LFR-3–8 and LFR-9–16, respectively. The stability of DANMF is worse when $\mu = 0.3$ than that when $\mu = 0.1$, which shows that its instability rises with the increase of network complexity. COPRA, MWLP and ComE have better stability than LinkLPA and DANMF but worse stability than other methods. Other methods obtain better stability than above algorithms, specifically, for LFR-3–9, the stability of GLPA is the first, DLPA$^+$ and NGLPA obtains suboptimal stability and GLLPA ranks the fourth. For LFR-10–16, the stability of GLPA is also the first, DLPA$^+$, WLPA and GLLPA gain the second, the third and the fourth rank, respectively, however, SLPA, LPA_NI and NGLPA obtain weaker stability than them. Although we fix the node order of label updating and the order of label launching and accepting, the overlapping nodes may be assigned to different sets in compact layout in large networks, which causes GLLPA still unstable. As a result, GLLPA obtains better stability than most methods and its standard deviation on the synthetic networks vibrates small.

In terms of Q_{OVavg}, the results of all methods decline more than those of NMI_{avg} when *mu* increases from 0.1 to 0.3, which show that Q_{OV} is more sensitive to mixing parameter *mu* than *NMI*. Meanwhile, the difference among Q_{OVavg} of methods is larger than that among NMI_{avg}. As shown in Fig. 6(e) and (f) and Table 7, GLLPA obtains the best Q_{OVavg} and the four methods (SLPA, WLPA, LPA_NI and NGLPA) gain suboptimal and similar results. They consider the weight of nodes and labels by the topological structure of networks or use node and label information, then they output good Q_{OVavg}. In particular, GLLPA utilizes extra layout information to initialize labels and uses the attractive forces among nodes to increase the closeness among nodes, thus, the Q_{OVavg} of GLLPA is the highest. We can find that the rest methods has poor Q_{OVavg}, especially, the results of COPRA falls fast when $mu = 0.1$. COPRA has good NMI_{avg}, however, the Q_{OVavg} of COPRA is poor in conformity with the conclusion reported in [68]. When $mu = 0.3$, the results of MWLP and DANMF are weaker than other methods.

Considering the standard deviation of Q_{OV} (Q_{OVstd}), as shown in Fig. 6(g) and (h), the stability of COPRA, LinkLPA, MWLP, ComE and DANMF is still worse than the rest methods for the same reason mentioned in Section 5.5.1. As shown in Table 7, DLPA$^+$ obtains the best stability for determining the node order of label updating by the confidence variance of nodes. NGLPA and GLPA obtain suboptimal stability, in which NGLPA considers the node weight by LeaderRank [51] to decide the updating order of nodes and GLPA constructs weighted networks to directly merge super nodes (i.e., node sets) to reduce randomness. WLPA, LPA_NI and GLLPA have similar stability in terms of Q_{OV}, while the stability of SLPA is weaker than them for randomly choosing nodes to update labels.

In terms of running time, as shown in Fig. 6(i) and (j) and Table 7, COPRA ranks the fifth because randomly choosing nodes can speed iterations, although this operation causes weak stability. SLPA discovers communities though all received labels, which requires more time to collect historical label information, thus, the running time of SLPA is inferior. The efficiency of DLPA$^+$ is worse than other methods except SLPA, ComE and DANMF. Its dominant labels cannot reflect the importance of labels in biggish networks well, then DLPA$^+$ needs more iterations. Meanwhile, DLPA$^+$ calculates the neighbor confidence and its variance to increase stability, which increases time cost. WLPA ranks the second because it simply updates node labels by the degree of nodes so as to save running time. LinkLPA obtains the third rank in biggish networks because it replaces the nodes to edges and assigns edges with labels to spread for detecting communities (the



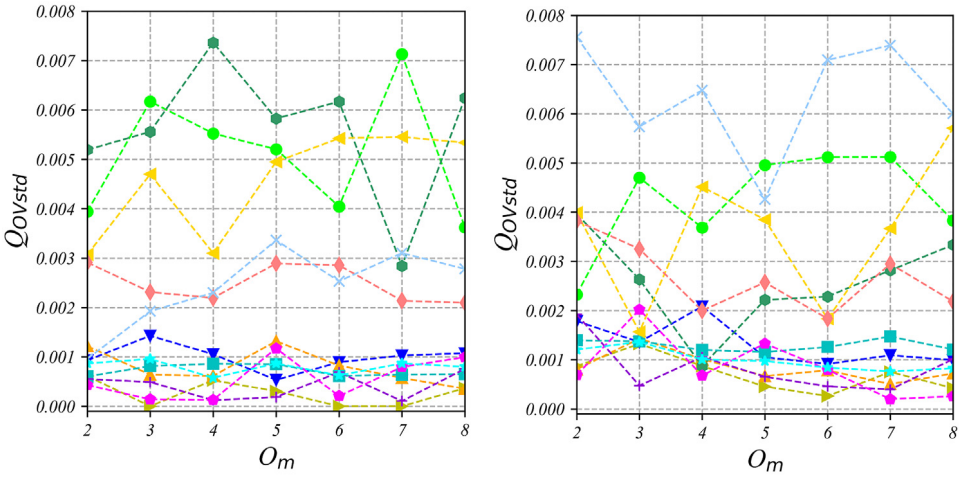
(a) $mu = 0.1$ (b) $mu = 0.3$

(c) $mu = 0.1$ (d) $mu = 0.3$

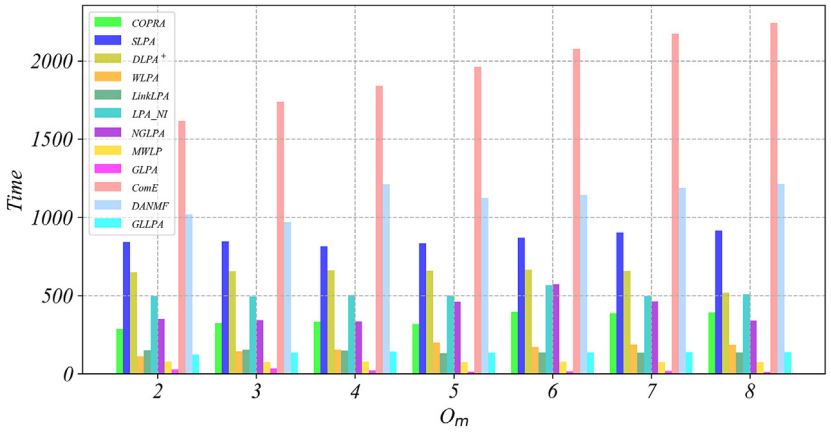
(e) $mu = 0.1$ (f) $mu = 0.3$

Fig. 6. Results for the synthetic networks with overlapping communities.

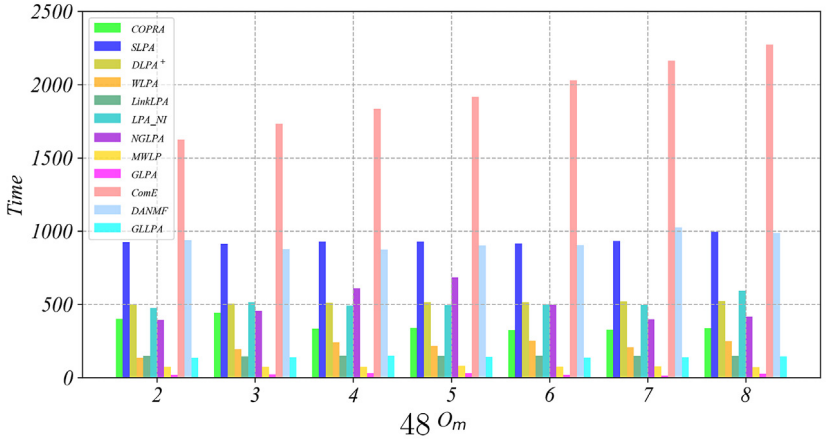
edge version of LPA), which is near linear time complexity. LPA_NI and NGLPA gain weak efficiency because the former computes the weight of nodes from the prior importance of nodes by Bayesian network and the latter iteratively computes the relations among nodes and their neighbors to calculate node weight by LeaderRank. MWLP ranks the second in terms of efficiency, however, it obtains poor NMI_{avg} and Q_{OVavg}, we consider that MWLP stops early and does not find the optimum. GLPA obtains the highest



(g) $mu = 0.1$ (h) $mu = 0.3$



(i) $mu = 0.1$



(j) $mu = 0.3$

Fig. 6. (*continued*).

efficiency in the networks with $n = 5000$, since it constructs label propagation graph to reduce edges and merges super nodes to operate many nodes once. GLLPA achieves the third efficiency. For the nodes, which should belong to the same communities, the label initialization strategy can make most of them have the same label at the beginning for speeding up convergence and node attraction can rope them to the same community in less iterations. Meanwhile, label influence can relatively accurately reflect the difference among labels and filter labels with small influence in the updating process. Although its running time is higher than MWLP and GLPA, its NMI_{avg} and Q_{OVavg} is the best. By analyzing the experimental results in synthetic networks, we consider that GLLPA needs less time to reveal desirable communities than state-of-the-art methods in most cases.

Table 7
Average rank for the synthetic networks with overlapping communities.

Criteria	Network	COPRA	SLPA	DLPA$^+$	WLPA	LinkLPA	LPA_NI	NGLPA	MWLP	GLPA	ComE	DANMF	GLLPA
NMI_{avg}	LFR-3–9	4.2857	5.5714	11.8571	3.4285	6.7142	4.5714	7.4285	10.5714	6.2857	6.2857	9.8571	**1.1428**
	LFR-10–16	2.7142	5.0000	10.7142	2.5714	5.0000	6.7142	7.7142	10.2857	6.4285	7.8571	12.0000	**1.0000**
NMI_{std}	LFR-3–9	9.0000	7.0000	2.8571	5.1428	11.8571	5.1428	3.1428	7.5714	**1.5714**	9.2857	11.1428	4.2857
	LFR-10–16	7.0000	7.0000	2.1428	3.4285	9.8571	5.2857	4.8571	10.5714	**2.0000**	9.5714	12.0000	4.2857
Q_{OVavg}	LFR-3–9	11.2857	5.4285	8.8571	3.4285	5.5714	4.0000	2.2857	11.4285	8.8571	7.7142	8.1428	**1.0000**
	LFR-10–16	7.1428	5.0000	7.4285	2.0000	9.5714	3.2857	3.7142	12.0000	9.2857	6.5714	11.0000	**1.0000**
Q_{OVstd}	LFR-3–9	11.0000	6.2857	**2.0000**	4.8571	11.4285	4.0000	2.5714	10.4285	3.4285	8.4285	8.5714	5.0000
	LFR-10–16	10.4285	5.7142	**2.2857**	3.5714	8.0000	6.2857	3.7142	9.7142	3.2857	8.8571	11.8571	4.2857
Time (s)	LFR-3–9	6.1428	10.0000	9.0000	4.5714	3.7142	7.8571	7.0000	2.0000	**1.0000**	12.0000	11.0000	3.7142
	LFR-10–16	6.1428	10.7142	8.4285	4.7142	4.1428	8.0000	7.42857	2.0000	**1.0000**	12.0000	10.2857	3.1428

5.6. Results for real-world networks

5.6.1. Results for the real-world networks with known communities

The results of *NMI*, Q_{OV} and running time for the real-world networks with known true communities are shown in Table 8. Here, we analyze the results of each algorithm according to their average rank.

- COPRA: The NMI_{avg} and Q_{OVavg} of COPRA are worse than other methods except MWLP and its running time is larger than other methods except SLPA, NGLPA, ComE and DANMF. In terms of stability, COPRA shows larger oscillation than most methods, but we can find that its standard deviation is smaller in biggish networks than that in small networks. COPRA updates the belonging coefficients of labels over all neighbor labels, then fewer neighbors in small networks may not reflect the community membership of nodes well. Thus, the stability of COPRA is sensitive in small networks. On the other hand, COPRA vibrates smaller in large networks, as some small communities can be merged to form large communities by the post-processing of COPRA to reduce randomness. However, due to randomly choose nodes to update, its stability is weak.
- SLPA: SLPA obtains the third NMI_{avg} and the fourth Q_{OVavg} with the lowest efficiency. Meanwhile, its stability is poor in small networks. SLPA can catch the latent relations among nodes by historical labels to identify node affiliation, then it obtains good NMI_{avg}. However, high NMI_{avg} may not lead to high Q_{OVavg} because true communities may be not the most dense in some networks [68], thus, its Q_{OVavg} is weak. SLPA randomly chooses nodes to update their labels, thus, its stability is poorer than most methods. We also find that the stability of SLPA in small networks is poorer than those in biggish networks, which is similar with COPRA. SLPA requires 100 time iterations to collect historical label information, then its running time is the highest.
- DLPA$^+$: DLPA$^+$ achieves the sixth NMI_{avg}, the third Q_{OVavg} and the fifth efficiency. In other words, it gets better performance than some methods in small networks, since it can deal with small networks by inflation factor and obtain fast convergence by propagating dominant labels. DLPA$^+$ obtains the best stability because it decides the node order of label updating by the confidence variance of nodes and chooses dominant labels to spread.
- WLPA: WLPA ranks the sixth in terms of NMI_{avg} and Q_{OVavg}, and its efficiency is the third. WLPA increases or decreases the weight of labels when the degree of nodes is larger or smaller than the degree of neighboring nodes. WLPA is more stable than some methods because it updates the nodes with large degree preferentially.
- LinkLPA: The NMI_{avg} and Q_{OVavg} of LinkLPA are worse than most methods. Meanwhile, its stability is worse. However, it achieves the second running time. LinkLPA uses edges instead of nodes to conduct label propagation. However, because a node will be grouped into one community even if there is only one edge belongs to this community, this transformation will lead to over-overlapping nodes. Although LinkLPA uses post-processing to handle over-overlapping, the operation cannot avoid this problem, then its results are poor but the running time is fast. LinkLPA randomly selects edges to update their labels and we know the number of edges is larger than the number of nodes in most networks, then the randomness of LinkLPA is large.
- LPA_NI: LPA_NI ranks the sixth for NMI_{avg} and running time, while it obtains the second rank in terms of Q_{OVavg}. LPA_NI uses a Bayesian model to learn the prior probability of node property and identify important nodes, which needs extra iterations. The stability of LPA_NI is poor in small networks because its node influence and label influence cannot reflect the accurate weight of labels in small networks.
- NGLPA: The NMI_{avg} and Q_{OVavg} of NGLPA are worse than most methods, which denotes that its node gravitation cannot reflect the true affiliation of nodes in small networks. In order to compute node gravitation, NGLPA computes node similarity and uses LeaderRank [51] to capture the relations among nodes and their neighbors iteratively, thus, it needs more running time. Its stability is worse than most algorithms in small networks. In small networks, the node gravitation of some labels may be the same easily for relatively simple network structure. Although NGLPA considers the node weight to decide the updating order of nodes, it randomly selects one label with the same node gravitation to update, which leads to obvious randomness in small networks.
- MWLP: The NMI_{avg} and Q_{OVavg} of MWLP are the worst, while it obtains the third running time. MWLP identifies the motif of networks and constructs a weighted network to update nodes' labels, which needs more running time, so we consider that MWLP stops early and obtains weak results. The stability of MWLP is mainly achieved by NaS voting strategy and the label propagation can proceed in a more stable way in small networks, then it obtains better stability than some methods in small networks.
- GLPA: GLPA obtains the best NMI_{avg} and the fifth Q_{OVavg}. GLPA makes the complementary entropy as objective function to maximize, which can detect near true communities, then its NMI_{avg} is competitive, especially 1.0000 for Dolphin network. However, the complementary entropy is not suitable for detecting dense groups, then its Q_{OVavg} is weak. Meanwhile, it achieves the highest efficiency. GLPA merge many nodes regarded as a super node to other super nodes to update the labels of multiple nodes once, then its running time is fast in small networks. In addition, it obtains good stability because GLPA operates many nodes once to reduce randomness.

Table 8
Results for the real-world networks with known communities.

Criterion	Network	COPRA	SLPA	DLPA^{+}	WLPA	LinkLPA	LPA_NI	NGLPA	MWLP	GLPA	ComE	DANMF	GLLPA
	Karate	0.3596 (12)	0.6915 (5)	0.5489 (7)	0.5016 (9)	0.5479 (8)	0.6598 (6)	0.4408 (11)	0.4468 (10)	0.7238 (4)	0.7445 (3)	**0.9836 (1)**	0.7532 (2)
	Dolphin	0.5976 (9)	0.6678 (6)	0.4753 (12)	0.6599 (7)	0.5453 (10)	0.6436 (8)	0.7108 (5)	0.4940 (11)	**1.0000 (1)**	0.9733 (2)	0.8759 (3)	0.7899 (4)
	Polbook	0.5556 (7)	0.5574 (5)	0.4993 (11)	0.5709 (3)	0.5558 (6)	0.5409 (8)	0.5353 (9)	0.3929 (12)	0.5222 (10)	0.5686 (4)	**0.6008 (1)**	0.5796 (2)
	Football	0.8836 (7)	0.8862 (6)	0.9044 (3)	0.9013 (4)	0.8504 (10)	0.8823 (8)	0.8887 (5)	0.7944 (11)	**0.9231 (1)**	0.8792 (9)	0.6972 (12)	0.9093 (2)
	Netscience	0.3566 (7)	0.3651 (4)	0.3858 (2)	0.3350 (12)	0.3586 (6)	0.3636 (5)	0.3470 (10)	0.3660 (3)	**0.4040 (1)**	0.3388 (11)	0.3521 (9)	0.3548 (8)
NMI_{avg}	Average rank	8.4	5.2	7.0	7.0	8.0	7.0	8.0	9.4	**3.4**	5.8	5.2	3.6
	Karate	0.0955 (8)	0.2184 (12)	**0.0000 (1)**	0.1427 (11)	0.0651 (6)	0.1192 (10)	0.0969 (9)	**0.0000 (1)**	0.0333 (3)	0.0532 (5)	0.0491 (4)	0.0661 (7)
	Dolphin	0.0987 (8)	0.1372 (11)	**0.0000 (1)**	0.0666 (7)	0.1032 (9)	0.1389 (12)	0.1147 (10)	0.0200 (3)	**0.0000 (1)**	0.0474 (6)	0.0464 (4)	0.0466 (5)
	Polbook	0.0343 (10)	0.0394 (11)	0.0082 (5)	0.0217 (7)	0.0525 (12)	0.0278 (9)	0.0232 (8)	0.0124 (6)	0.0034 (3)	0.0022 (2)	**0.0009 (1)**	0.0074 (4)
	Football	0.0244 (9)	0.0178 (6)	0.0098 (3)	0.0180 (7)	0.0536 (11)	0.0272 (10)	0.0121 (4)	**0.0039 (1)**	0.0154 (5)	0.0232 (8)	0.0635 (12)	0.0076 (2)
	Netscience	0.0018 (8)	0.0021 (10)	0.0005 (2)	0.0014 (6)	0.0019 (9)	0.0012 (5)	0.0016 (7)	0.0006 (3)	0.0008 (4)	0.0052 (12)	0.0035 (11)	**0.0002 (1)**
NMI_{std}	Average rank	8.6	10.0	**2.4**	7.6	9.4	9.2	7.6	2.8	3.2	6.6	6.4	3.8
	Karate	0.2348 (12)	0.3742 (7)	0.4210 (3)	0.3682 (8)	0.3199 (10)	0.4136 (5)	0.3314 (9)	0.2429 (11)	0.4261 (2)	0.3897 (6)	0.4201 (4)	**0.4387 (1)**
	Dolphin	0.3741 (11)	0.4757 (5)	0.5166 (2)	0.3695 (12)	0.3998 (7)	0.5055 (3)	**0.5189 (1)**	0.4119 (6)	0.3948 (10)	0.3961 (9)	0.3997 (8)	0.4962 (4)
	Polbook	0.4884 (7)	0.4923 (6)	0.5249 (2)	0.5070 (4)	0.4606 (10)	0.5048 (5)	0.4600 (11)	0.3522 (12)	**0.5267 (1)**	0.4704 (8)	0.4626 (9)	0.5071 (3)
	Football	0.5972 (6)	0.6016 (2)	0.5960 (7)	0.5981 (4)	0.5835 (9)	0.5985 (3)	0.5848 (8)	0.2829 (12)	0.5981 (4)	0.5640 (10)	0.5318 (11)	**0.6089 (1)**
	Netscience	0.8784 (8)	0.9043 (6)	0.8456 (9)	0.9279 (2)	0.8785 (7)	0.9140 (4)	0.9209 (3)	0.9066 (5)	0.7252 (12)	0.8203 (10)	0.7822 (11)	**0.9283 (1)**
Q_{OVavg}	Average rank	8.8	5.2	4.6	6.0	8.6	4.0	6.4	9.2	5.8	8.6	8.6	**2.0**
	Karate	0.1018 (12)	0.0868 (7)	**0.0000 (1)**	0.0817 (10)	0.0129 (4)	0.0659 (9)	0.0366 (7)	**0.0000 (1)**	0.0555 (8)	0.0240 (6)	0.0034 (3)	0.0139 (5)
	Dolphin	0.0394 (9)	0.0554 (12)	**0.0000 (1)**	0.0251 (7)	0.0258 (8)	0.0486 (10)	0.0493 (11)	0.0226 (6)	**0.0000 (1)**	0.0022 (3)	0.0029 (4)	0.0071 (5)
	Polbook	0.0321 (11)	0.0213 (10)	0.0041 (4)	0.0062 (5)	0.0332 (12)	0.0202 (9)	0.0102 (7)	0.0164 (8)	0.0004 (2)	**0.0002 (1)**	0.0012 (3)	0.0069 (6)
	Football	0.0211 (8)	0.0107 (3)	0.0093 (2)	0.0137 (6)	0.0354 (11)	0.0130 (5)	0.0179 (7)	0.0125 (4)	0.0381 (12)	0.0222 (9)	0.0314 (10)	**0.0066 (1)**
	Netscience	0.0184 (10)	0.0068 (7)	0.0032 (4)	0.0065 (6)	0.0073 (8)	0.0023 (2)	0.0026 (3)	0.0048 (5)	0.0121 (9)	0.0385 (12)	0.0336 (11)	**0.0002 (1)**
Q_{OVstd}	Average rank	10	8.6	**2.4**	6.8	8.6	7.0	7.0	4.8	6.4	6.2	6.2	3.6
	Karate	0.1788 (8)	6.4314 (12)	0.0071 (2)	0.1219 (7)	0.0354 (4)	0.1024 (6)	0.1788 (8)	0.0774 (5)	**0.0004 (1)**	2.6866 (11)	1.5753 (10)	0.0108 (3)
	Dolphin	0.4777 (9)	13.3722 (12)	0.3522 (6)	0.4155 (8)	0.0769 (2)	0.3485 (5)	0.3527 (7)	0.1444 (3)	**0.0028 (1)**	6.2121 (11)	2.1510 (10)	0.3028 (4)
	Polbook	1.0289 (8)	35.3845 (12)	0.8963 (6)	0.8330 (5)	0.2861 (2)	1.1117 (9)	1.0010 (7)	0.3343 (3)	**0.0065 (1)**	7.3902 (11)	3.5580 (10)	0.8294 (4)
	Football	1.0073 (8)	48.5151 (12)	0.6431 (6)	0.5485 (5)	0.4858 (2)	0.6656 (7)	13.9052 (11)	0.5075 (3)	**0.0255 (1)**	11.6591 (10)	9.4869 (9)	0.5381 (4)
	Netscience	33.2918 (8)	245.5020 (11)	5.8815 (4)	23.5540 (7)	2.7980 (2)	17.4400 (6)	73.9827 (10)	6.1167 (5)	**0.2409 (1)**	257.6490 (12)	46.1743 (9)	5.3820 (3)
Time (s)	Average rank	8.2	11.8	4.8	6.4	2.4	6.6	8.6	3.8	**1.0**	11.0	9.6	3.6

Table 9
Results for the real-world networks without known communities.

Criterion	Network	COPRA	SLPA	DLPA$^+$	WLPA	LinkLPA	LPA_NI	NGLPA	MWLP	GLPA	GLLPA
Q_{OVavg}	Lesmis	0.4846 (6)	0.5397 (4)	0.5446 (3)	0.5515 (2)	0.4509 (8)	0.5262 (5)	0.4477 (9)	0.4762 (7)	0.4037 (10)	0.5561 (1)
	Jazz	0.3954 (3)	0.3457 (5)	0.2821 (7)	0.3624 (4)	0.2832 (6)	0.4343 (2)	0.2812 (8)	0.2474 (9)	0.2175 (10)	**0.4397 (1)**
	Email	0.2915 (2)	0.2905 (4)	0.2401 (9)	0.2908 (3)	0.2453 (8)	0.2726 (6)	0.2734 (5)	0.2348 (10)	0.2703 (7)	**0.2917 (1)**
	Power	0.1696 (10)	0.6225 (7)	0.5893 (9)	0.7731 (3)	0.5935 (8)	0.7473 (5)	0.7631 (4)	0.7454 (6)	0.7773 (2)	**0.7840 (1)**
	PGP	0.5117 (10)	0.7641 (5)	0.6761 (7)	0.6231 (9)	0.6545 (8)	0.7861 (4)	**0.8092 (1)**	0.7878 (2)	0.6831 (6)	0.7865 (3)
	Cond2003	0.6306 (4)	0.6341 (2)	0.4764 (7)	0.5959 (5)	0.4738 (8)	0.6313 (3)	0.5907 (6)	0.4241 (9)	0.3993 (10)	**0.6501 (1)**
	Enron	0.3068 (6)	0.3140 (4)	0.2844 (7)	**0.3191 (1)**	0.3089 (5)	0.3182 (2)	0.2728 (8)	0.2259 (10)	0.2555 (9)	0.3181 (3)
	Cond2005	0.4256 (7)	0.6019 (4)	0.4371 (6)	0.6117 (2)	0.4089 (9)	0.6111 (3)	0.4431 (5)	0.4224 (8)	0.3710 (10)	**0.6187 (1)**
	DBLP	0.6638 (3)	0.6655 (2)	0.5075 (8)	0.6627 (4)	0.6293 (6)	0.6324 (5)	0.6162 (7)	0.4391 (10)	0.4556 (9)	**0.6926 (1)**
	Amazon	0.7699 (3)	0.7405 (6)	0.6415 (9)	0.7690 (4)	0.7455 (5)	0.7714 (2)	0.7137 (7)	0.6091 (10)	0.6555 (8)	**0.7932 (1)**
	Average Rank	5.4	4.3	7.2	3.7	7.1	3.7	6.0	8.1	8.1	**1.4**
Q_{OVstd}	Lesmis	0.0468 (7)	0.0365 (5)	**0.0006 (1)**	0.0402 (6)	0.0659 (9)	0.0627 (8)	0.0815 (10)	0.0080 (3)	0.0158 (4)	0.0064 (2)
	Jazz	0.0713 (7)	0.0916 (9)	**0.0000 (1)**	0.0804 (8)	0.1094 (10)	0.0363 (5)	0.0463 (6)	0.0134 (4)	0.0084 (3)	0.0011 (2)
	Email	0.0256 (10)	0.0072 (5)	0.0022 (3)	0.0242 (9)	0.0173 (6)	0.0191 (7)	0.0215 (8)	0.0003 (2)	**0.0000 (1)**	0.0062 (4)
	Power	0.0089 (9)	0.0039 (6)	**0.0017 (1)**	0.0037 (5)	0.0040 (8)	0.0034 (4)	0.0019 (2)	0.0021 (3)	0.0362 (10)	0.0039 (6)
	PGP	0.0955 (10)	0.0059 (7)	0.0014 (2)	0.0322 (9)	0.0125 (8)	0.0039 (6)	0.0020 (3)	0.0030 (4)	0.0034 (5)	**0.0012 (1)**
	Cond2003	0.0039 (7)	0.0029 (5)	0.0032 (6)	0.0055 (8)	0.0023 (4)	0.0014 (3)	0.0097 (9)	0.1488 (10)	**0.0005 (1)**	0.0013 (2)
	Enron	0.0007 (6)	0.0088 (10)	0.0002 (3)	0.0005 (4)	0.0052 (8)	0.0034 (7)	0.0001 (2)	0.0060 (9)	**0.0000 (1)**	0.0005 (4)
	Cond2005	0.0039 (7)	0.0027 (5)	0.0015 (3)	0.0194 (10)	0.0020 (4)	0.0086 (9)	0.0040 (8)	0.0033 (6)	**0.0003 (1)**	0.0006 (2)
	DBLP	0.0026 (7)	0.0065 (10)	0.0005 (4)	0.0034 (8)	0.0022 (6)	0.0001 (1)	0.0008 (5)	0.0046 (9)	0.0002 (3)	**0.0001 (1)**
	Amazon	0.0053 (9)	0.0022 (5)	0.0052 (8)	0.0008 (4)	0.0026 (6)	0.0005 (2)	0.0078 (10)	0.0036 (7)	0.0005 (2)	**0.0001 (1)**
	Average Rank	7.9	6.7	3.2	7.1	6.9	5.2	6.3	5.7	3.1	**2.5**
Time (s)	Lesmis	0.4285 (8)	19.5511 (10)	0.2447 (4)	0.3132 (6)	0.1416 (2)	0.4174 (7)	4.5464 (9)	0.2316 (3)	**0.0020 (1)**	0.2799 (5)
	Jazz	4.1365 (6)	201.9469 (10)	2.6987 (3)	4.4134 (8)	4.3000 (7)	3.3010 (4)	46.7322 (9)	1.6083 (2)	**0.0636 (1)**	3.8159 (5)
	Email	143.7178 (9)	415.8542 (10)	109.7681 (8)	22.2821 (5)	13.4471 (3)	72.2570 (6)	102.1098 (7)	5.9799 (2)	**3.4327 (1)**	22.0687 (4)
	Power	341.5182 (9)	211.0747 (8)	167.6967 (7)	145.2718 (6)	**18.3522 (1)**	138.3604 (5)	92.9416 (4)	34.8291 (2)	588.9048 (10)	89.8743 (3)
	PGP	3909.1786 (10)	767.6998 (8)	518.1078 (6)	2034.5744 (9)	253.2977 (3)	558.5890 (7)	390.1165 (4)	**230.4116 (1)**	468.5890 (5)	247.8083 (2)
	Cond2003	10471.5065 (10)	4624.2670 (9)	3997.4615 (8)	3560.0126 (7)	919.0770 (3)	2090.6047 (4)	2316.1850 (5)	2649.2855 (6)	894.1633 (2)	**688.7012 (1)**
	Enron	5893.1192 (8)	6197.1434 (9)	6283.5557 (10)	2563.4053 (4)	1262.1980 (3)	3112.7570 (5)	3666.8810 (6)	3916.5000 (7)	**907.9960 (1)**	1218.6368 (2)
	Cond2005	24132.8245 (10)	6206.6146 (8)	6705.6510 (9)	4883.3493 (6)	1563.8270 (3)	3480.0526 (4)	3853.6380 (5)	5722.0620 (7)	**824.4700 (1)**	1003.4027 (2)
	DBLP	250806.2440 (10)	243767.2435 (9)	22444.7200 (5)	11435.1134 (3)	26459.6173 (6)	20281.4485 (4)	32492.8411 (7)	52261.1627 (8)	12525.9250 (2)	**11294.6472 (1)**
	Amazon	377662.4635 (10)	360883.7190 (9)	35257.9160 (6)	13168.8880 (2)	30547.2516 (5)	22555.8957 (4)	38266.2185 (7)	58619.9457 (8)	15369.4360 (3)	**12841.3391 (1)**
	Average Rank	9.0	9.0	6.6	5.6	3.6	5.0	6.3	4.6	2.7	**2.6**

- ComE: ComE obtains the fourth NMI_{avg} but weak Q_{OVavg} with slow running time. We can find that the accuracy of NMI_{avg} and Q_{OVavg} is good in small networks and decreases with the increase of network scale, which may because node embeddings can capture the relations among nodes in small networks, while they cannot catch more features in biggish networks. In Dolphin network, it obtains the NMI_{avg} of 0.9733, which is close to the optimum and shows that small community number is beneficial to capture the high order community feature of community embeddings in ComE. Its stability is middle, since the embeddings can contain enough features of small networks to help clustering methods to detect communities.
- DANMF: DANMF obtains the third NMI_{avg} but weak Q_{OVavg} with slow running time. Similar with ComE, the accuracy of NMI_{avg} and Q_{OVavg} is good in small networks and decreases with the increase of network scale, which may because deep matrix factorization can catch the multilevel mappings between the original networks and community partitions in small networks, while it cannot catch complete mappings in biggish networks. Increasing the number of hidden layers may improve the accuracy. In Karate network, it obtains the NMI_{avg} of 0.9836, which is close to the optimum. Its stability is middle in small networks, since deep matrix factorization can contain enough hidden features of small networks to help clustering methods to detect communities.
- GLLPA: GLLPA obtains the second NMI_{avg}, the best Q_{OVavg} with high efficiency. The label initialization strategy and node attraction can make GLLPA detect accurate communities with less time. Its results are more stable than most algorithms because GLLPA decides the node order of label updating in the ascending order of node influence, the order of label launching in the descending order of belonging coefficient and the order of label accepting in the descending order of label influence. However, we find GLLPA still unstable, since it randomly selects one of the labels to assign to other nodes when the number of the most influential labels of a node is more than one. Meanwhile, overlapping nodes may gather to different sets in the compact layout of a network, then the label initialization result of community detection is different. Thus, GLLPA also has the instability problem. Overall, the stability of GLLPA is superior.

5.6.2. Results for the real-world networks without known communities

The results of Q_{OV} and running time for the real-world networks with unknown true communities are shown in Table 9. Because ComE and DANMF need the number of communities in networks as priori knowledge, then we do not compare them in the real-world networks without know communities.

In terms of Q_{OVavg}, as shown in Table 9, comparing with the best baseline, GLLPA improves Q_{OV} from 0.5515, 0.4343, 0.2915, 0.7773, 0.6341, 0.6117, 0.6655 and 0.7714 to 0.5561, 0.4397, 0.2917, 0.7840, 0.6501, 0.6187, 0.6926 and 0.7932 in Lesmis, Jazz, Email, Power, Cond2003, Cond2005, DBLP and Amazon networks, respectively, which are promoted by 0.46%, 0.54%, 0.02%, 0.67%, 1.60%, 0.70%, 2.71% and 2.18%, respectively. We can find that Q_{OV} is improved more in large networks (e.g., DBLP and Amazon networks) than in small networks. Traditional label initialization assigns each node with a unique label, which will produce many different labels in large networks at the beginning. For GLLPA, label initialization strategy based on the compact layout can assign nodes locating at the same position (i.e., belong to the same community possibly) with the same label, which can reduce redundant labels in large networks and increases accuracy. GLLPA obtains the best result according to the average rank, specially, GLLPA obtains the best Q_{OVavg} in most large networks. However, GLLPA is poorer than some methods in some networks (e.g., WLPA in Enron network and NGLPA in PGP network). Among baselines, the Q_{OVavg} of MWLP and GLPA is the worst.

In terms of stability, COPRA, SLPA, WLPA and LinkLPA are inferior. Among them, COPRA, SLPA and LinkLPA randomly choose nodes or edges to update, WLPA only chooses nodes according to their degree, then their stability are weak. DLPA$^+$ ranks the third because it fixes the node order of label updating by the confidence variance of nodes and chooses dominant labels to propagate. Analyzing the number of nodes in networks, we can find that its stability slightly falls with the increase of network scale, whose reason may be that the inflation factor of DLPA$^+$ cannot adjust the overlapping rate of nodes well in large networks. LPA_NI, NGLPA and MWLP obtain slightly better stability, since LPA_NI and NGLPA considers the weight of nodes and labels to determine the updating and propagating order. MWLP uses NaS strategy to propagate labels by both the number and strength of edges, which can obtain more stable results in small networks, however, this strategy exists more randomness in large networks. GLPA obtains the second stability for the same reason mentioned above. GLLPA obtains the best stability and its stability becomes better in large networks because label initialization strategy also can reduce randomness for making most nodes should in the same community with the same label instead of accepting from neighbors in subsequent iterations, which plays better in large networks.

Considering the running time, COPRA and DLPA$^+$ shows the deteriorating trend with the increase of network scale, while SLPA and NGLPA present an ameliorating trend. The idea of WLPA that simply increasing or decreasing the weight of labels according to node degree saves running time and gains nice results in large networks. On the contrary, the performance of DLPA$^+$ drops in large networks. The dominant label cannot adapt to large networks well possibly and makes DLPA$^+$ need more running time, which is same as its performance in synthetic networks. LinkLPA simple updates the labels of edges over all neighboring edges, which can be seen as the edge updating version of LPA, then its efficiency is high. The results of LPA_NI and NGLPA are similar with those in the real-world networks with known communities, where the former computes the weight of nodes by Bayesian network and the latter calculates node weight by LeaderRank, which need some time but their operations can reduce iteration times for identifying the community membership of nodes in large networks, then their efficiency is improved in large networks. We have observed that the running time of WMLP increases with the increase of network scales because finding the motif of networks needs more time in large networks for time complexity $O(n^3)$. GLPA also achieves better efficiency than other methods except GLLPA. We consider that computing node similarity requires more time in large networks, then its efficiency is poorer than GLLPA, but merging many nodes once can reduce running time, then its efficiency is higher than most methods. Thanks to the effectiveness of label initialization strategy and label influence, GLLPA achieves the best average rank in terms of running time in these networks, especially in large networks. Label initialization strategy can assign most nodes should in the same community with the same label to reduce iteration time. Label influence can help model to filter unimportant labels to accelerate convergence. Overall, GLLPA outperforms state-of-the-art baselines in most real-world networks.

5.7. Results of layout energy

As shown in Table 10, we present the average energy of nodes ($\varepsilon(p|G)/n$) of compact layout and uniform layout in networks

Table 10
The energy for compact layout and uniform layout of networks.

Network	Compact layout	Uniform layout	Network	Compact layout	Uniform layout
LFR-1	−5953	−4365	Karate	−450	−364
LFR-2	−6155	−4573	Dolphin	−465	−381
LFR-3	−5952	−4400	Polbook	−2514	−2035
LFR-4	−5938	−4376	Football	−2613	−2064
LFR-5	−5989	−4445	Netscience	−443	−280
LFR-6	−5920	−4406	Lesmis	−1905	−1608
LFR-7	−5893	−4362	Jazz	−97591	−68607
LFR-8	−5767	−4299	Email	−7702	−5874
LFR-9	−5926	−4421	Power	−31	−21
LFR-10	−6238	−4629	PGP	−5297	−3521
LFR-11	−6161	−4612	Cond2003	−11329	−8031
LFR-12	−6209	−4648	Enron	−345224	−239592
LFR-13	−6213	−4621	Cond2005	−20860	−14690
LFR-14	−6151	−4617	DBLP	−11142	−7840
LFR-15	−6076	−4535	Amazon	−14441	−11951
LFR-16	−6160	−4567			

obtained by GLLPA. (1) [The energy of compact layout] GLLPA first conducts compact layout. When compact layout reaches stop condition (i.e., energy does not decrease), GLLPA begins to perform uniform layout and community detection. Thus, the energy of compact layout should reach the optimum (local). (2) [The energy of uniform layout] GLLPA is used to discover communities in networks and uniform layout is used to assist community detection, then GLLPA stops when community detection part reaches termination condition, and the uniform layout is also stopped and may not reach the optimum.

We can find that (1) the energy of the two layout is negative because the force is the negative gradient of energy [39]. When the layout reaches equilibrium state, the resultant force of all nodes in networks should equal to zero, but each node may be acted on non-zero forces, then the total energy, which can be seen as the sum of the energy of each node in networks, is negative. (2) the energy of uniform layout is higher than the energy of compact layout, which also shown that compact layout reaches more closely to the optimal value than uniform layout as mentioned above. (3) the average energy may be related to the average degree of networks, since node attraction and node repellent are produced among adjacent nodes. If the average degree is large, then a node will suffer more forces from its neighbors, then the energy of networks is small. For example, in synthetic networks, their average degree equals to 10, then their energies are close. Only the energies of networks when $mu = 0.3$ are slightly smaller than that when $mu = 0.1$. In real-world networks, the energies of Jazz and Enron networks are much smaller than others because their average degree is higher than others. The energy of Power network is higher than others because its average degree is the smallest, then the nodes of Power network suffer less forces and produce higher energy.

6. Conclusion

In this paper, a graph layout based label propagation algorithm GLLPA is proposed for detecting communities, which uses network layout information to assist community detection. A label initialization strategy is designed to speed up its convergence, and node attraction, node influence and label influence are proposed to handle the randomness and enhance the accuracy and efficiency. Experimental results on 16 synthetic and 15 real-world networks demonstrate that GLLPA can provide communities better and sooner than other algorithms in most cases.

We also find that GLLPA is still unstable, and NMI_{avg} and Q_{OVavg} obtained by GLLPA are poorer than some algorithms in some networks. In the future, we plan to improve node attraction, node influence and label influence to detect accurate communities and improve the instability. For example, if the number of the most important labels of a node is more than one, we can define a second index to further distinguish the weight of labels or use a label memory to store them [69]. The community detection part of GLLPA uses the information of network layout, however, the layout part does not utilize the information of community detection. Designing new combined ways or joint learning strategies of graph layout and community detection is also an important research area. Also, we are interested in exploring attribute networks [70] by introducing node influence and label influence in future work.

CRediT authorship contribution statement

Yun Zhang: Conceptualization, Methodology, Software, Writing - original draft. **Yongguo Liu:** Conceptualization, Funding acquisition, Writing - review & editing. **Rongjiang Jin:** Investigation, Resources, Software. **Jing Tao:** Visualization, Validation. **Lidian Chen:** Validation, Supervision. **Xindong Wu:** Validation, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was supported in part by the National Key R&D Program of China under grants 2019YFC1710300 and 2017YFC1703905, and the Sichuan Science and Technology Program under grants 2020YFS0283, 2020YFS0302 and 2020YFS0372.

Appendix

Details of Compact Layout for Label Initialization Strategy

The proposed label initialization strategy is based on the results of compact layout. After GLLPA finishes compact layout, we initiate nodes owning the same position with the same label to increase accuracy and reduce iteration time of community detection part. Ideally, the nodes of the same community should collapse into a single position in compact layout [33], then we use (a, r)-energy model with $a = -0.95$ and $r = -1$ proposed in [33] to draw networks to compact layout, in which nodes of the same community tend to gather into a single position under large node attraction. Here, we reference the interpretations in [33] about that why choosing $a = -0.95$ and $r = -1$ can increase node

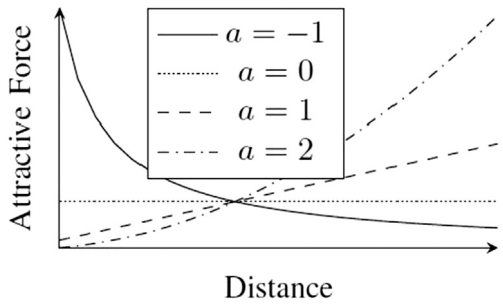


Fig. 7. Attractive force depending on a [33].

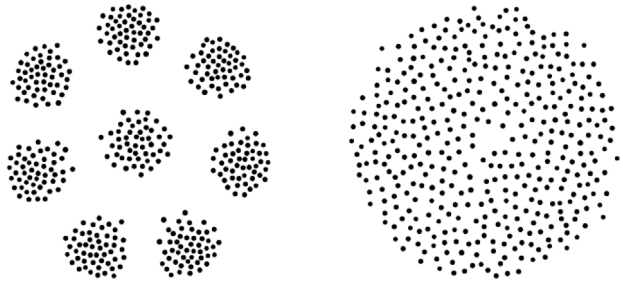


Fig. 8. Layouts with LinLog energy ($a - r = 1$) and with Fruchterman–Reingold energy ($a - r = 3$) of a pseudorandom network with eight clusters [39].

attraction and gather nodes in the same community together as follows.

Fig. 7 shows the trend of node attraction depending on the value of a. When $a = -1$, node attraction grows exponentially as the distance approaches zero. When larger values are used, it becomes weaker to prevent nodes from overlapping with each other. The energy model should meet the condition that $a > r$ to ensure that the attractive force between connected nodes grows faster than the repulsive force, and thus prevents infinite distances except between unconnected components [39], then a should be greater than $r = -1$. Thus, it is natural to choose a value slightly larger than -1, which is -0.95. (According to the experiments of [33], the difference between $a = -0.95$ and $a = -0.99$ is negligible.)

The value of r is set to -1 because it is a typical value for many force-directed graph layout algorithms, such as LinLog algorithm [71], ForceAtlas algorithm [72] and Fruchterman–Reingold algorithm [35]. The distances between communities are less dependent on densities for large a - r [39], in other words, the energy model with small $a - r$ can obtain layout with more obvious clustering tendency. For example, as shown in Fig. 8, LinLog algorithm obtains obvious clusters, while Fruchterman–Reingold obtains uniform layout because the $a - r = 1$ of the former is smaller than the a - $r = 3$ of the latter [39]. Then we need small $a - r$ for gathering the nodes of the same community together. Meanwhile, according to the condition mentioned above, r should be less than $a = -0.95$. Thus, it is preferable to choose a value smaller than $a = -0.95$ and as large as possible, which is -1.

Lim et al. [33] examined the clustering tendency while varying a when r is fixed to -1. Hopkins statistic can evaluate how far away a data set is from being uniformly distributed in the data space [73]. The closer Hopkins statistic is to 0, the higher the clustering tendency is, which shows that nodes tend to gather together. They generated a set of 24 various graphs by changing the parameters of the LFR benchmark and measured the Hopkins statistic of the layout results for the set of the graphs for each value of a. As shown in the box plot of Fig. 9 [33], the Hopkins

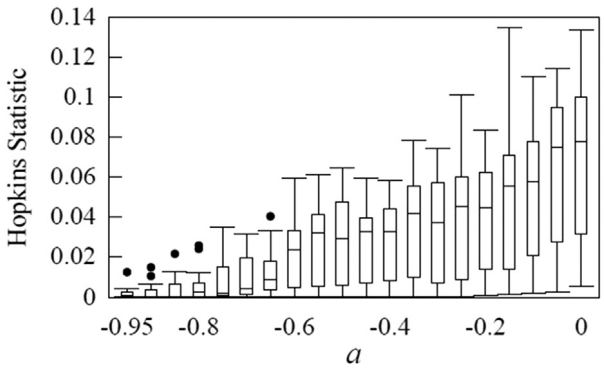


Fig. 9. Hopkins statistic depending on a [33].

statistic tends to decrease and becomes almost 0 when $a = -0.95$, which confirms that $a = -0.95$ and $r = -1$ can increase the node attraction among nodes to gather nodes together. Thus, we propose a new label initialization strategy by setting (a, r)-energy model with $a = -0.95$ and $r = -1$ to collapse nodes and assign the nodes locating in a single position with the same label to reduce the iteration time of community detection.

References

[1] T. Chakraborty, S. Ghosh, N. Park, Ensemble-based overlapping community detection using disjoint community structures, Knowl.-Based Syst. 163 (2019) 241–251, http://dx.doi.org/10.1016/j.knosys.2018.08.033.

[2] L. Yang, X.C. Cao, D.X. He, C. Wang, W.X. Zhang, Modularity based community detection with deep learning, in: Proceedings of International Joint Conference on Artificial Intelligence, 2016, pp. 2252–2258, http://dx.doi.org/10.5555/3060832.3060936.

[3] X.M. You, Y.H. Ma, Z.Y. Liu, A three-stage algorithm on community detection in social networks, Knowl.-Based Syst. 187 (2020) 104822, http://dx.doi.org/10.1016/j.knosys.2019.06.030.

[4] J.P. Zhang, X.Y. Ding, J. Yang, Revealing the role of node similarity and community merging in community detection, Knowl.-Based Syst. 165 (2019) 407–419, http://dx.doi.org/10.1016/j.knosys.2018.12.009.

[5] M.L. Lu, Z.L. Zhang, Z.H. Qu, Y. Kang, LPANNI: Overlapping community detection using label propagation in large-scale complex networks, IEEE Trans. Knowl. Data Eng. 31 (9) (2019) 1736–1749, http://dx.doi.org/10.1109/TKDE.2018.2866424.

[6] M. Kitsak, L.K. Gallos, S. Havlin, F. Liljeros, L. Muchnik, H.E. Stanley, H.A. Makse, Identification of influential spreaders in complex networks, Nature Phys. 6 (11) (2010) 888–893, http://dx.doi.org/10.1038/nphys1746.

[7] Y.X. Zhao, S.H. Li, J. Feng, Identification of influential nodes in social networks with community structure based on label propagation, Neurocomputing 210 (2016) 34–44, http://dx.doi.org/10.1016/j.neucom.2015.11.125.

[8] M.E.J. Newman, Fast algorithm for detecting community structure in networks, Phys. Rev. E 69 (6) (2004) 066133, http://dx.doi.org/10.1103/PhysRevE.69.066133.

[9] N.F. Haq, M. Moradi, Z. Wang, Community structure detection from networks with weighted modularity, Pattern Recognit. Lett. 122 (2019) 14–22, http://dx.doi.org/10.1016/j.patrec.2019.02.005.

[10] J.R. Zhu, B.L. Chen, Y.F. Zeng, Community detection based on modularity and k-plexes, Inform. Sci. 513 (2020) 127–142, http://dx.doi.org/10.1016/j.ins.2019.10.076.

[11] L. Bai, J.Y. Liang, H.Y. Du, Y.K. Guo, A novel community detection algorithm based on simplification of complex networks, Knowl.-Based Syst. 143 (2018) 58–64, http://dx.doi.org/10.1016/j.knosys.2017.12.007.

[12] X. Huang, Y. Ye, H. Guo, Y. Cai, H. Zhang, Y. Li, DSKmeans: A new kmeans-type approach to discriminative subspace clustering, Knowl.-Based Syst. 70 (2014) 293–300, http://dx.doi.org/10.1016/j.knosys.2014.07.009.

[13] S. Hirai, K. Yamanishi, Correction to efficient computation of normalized maximum likelihood codes for Gaussian mixture models with its applications to clustering, IEEE Trans. Inform. Theory 65 (10) (2019) 6827–6828, http://dx.doi.org/10.1109/TIT.2019.2915237.

[14] J.R. Xie, S. Kelley, B.K. Szymanski, Overlapping community detection in networks: The state-of-the-art and comparative study, ACM Comput. Surv. 45 (4) (2013) 43, http://dx.doi.org/10.1145/2501654.2501657.

[15] W.M. Li, S. Jiang, Q. Jin, Overlap community detection using spectral algorithm based on node convergence degree, Future Gener. Comput. Syst. 79 (2018) 408–416, http://dx.doi.org/10.1016/j.future.2017.08.028.

[16] I. Psorakis, S. Roberts, M. Ebden, B. Sheldon, Overlapping community detection using Bayesian non-negative matrix factorization, Phys. Rev. E 83 (2011) 066114, http://dx.doi.org/10.1103/PhysRevE.83.066114.

[17] F.H. Ye, C. Chen, Z.B. Zheng, Deep autoencoder-like nonnegative matrix factorization for community detection, in: Proceedings of ACM International Conference on Information and Knowledge Management, 2018, pp. 1393–1402, http://dx.doi.org/10.1145/3269206.3271697.

[18] S. Cavallari, V.W. Zheng, H. Cai, K.C.-C. Chang, E. Cambria, Learning community embedding with community detection and node embedding on graphs, in: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 2017, pp. 377–386, http://dx.doi.org/10.1145/3132847.3132925.

[19] F.Z. Liu, S. Xue, J. Wu, C. Zhou, H.W. Bin, C. Paris, S. Nepal, J. Yang, P.S. Yu, Deep learning for community detection: Progress, challenges and opportunities, 2020, arXiv:arXiv.2005.08225.

[20] H.Y. Cai, V.W. Zheng, K.C. Chang, A comprehensive survey of graph embedding: Problems, techniques and applications, 2017, arXiv:arXiv.1709.07604.

[21] M. Shen, Z. Ma, A novel node gravitation-based label propagation algorithm for community detection, Internat. J. Modern Phys. C 30 (6) (2019) 1950049, http://dx.doi.org/10.1142/S0129183119500499.

[22] Z.H. Deng, H.H. Qiao, Q. Song, L. Gao, A complex network community detection algorithm based on label propagation and fuzzy C-means, Physica A 519 (2019) 217–226, http://dx.doi.org/10.1016/j.physa.2018.12.024.

[23] P.Z. Li, L. Huang, C.D. Wang, J.H. Lai, D. Huang, Community detection by motif-aware label propagation, ACM Trans. Knowl. Discov. Data 14 (2) (2019) http://dx.doi.org/10.1145/3378537.

[24] U.N. Raghavan, R. Albert, S.R.T. Kumara, Near linear time algorithm to detect community structures in large-scale networks, Phys. Rev. E 76 (3) (2007) 036106, http://dx.doi.org/10.1103/PhysRevE.76.036106.

[25] H.L. Sun, J. Liu, J.B. Huang, G.T. Wang, X.L. Jia, Q.B. Song, LinkLPA: A link-based label propagation algorithm for overlapping community detection in networks, Comput. Intell. 33 (2) (2017) 308–331, http://dx.doi.org/10.1111/coin.12087.

[26] G. Yang, W.P. Zheng, C.H. Che, W.J. Wang, Graph-based label propagation algorithm for community detection, Int. J. Mach. Learn. Cybern. 11 (6) (2019) 1319–1329, http://dx.doi.org/10.1007/978-3-030-05411-3_14.

[27] S. Gregory, Finding overlapping communities in networks by label propagation, New J. Phys. 12 (10) (2010) 103018, http://dx.doi.org/10.1088/1367-2630/12/10/103018.

[28] J.R. Xie, B.K. Szymanski, X.M. Liu, SLPA: Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process, in: Proceedings of IEEE International Conference on Data Mining Workshops, 2011, pp. 344–349, http://dx.doi.org/10.1109/ICDMW.2011.154.

[29] Y. Xing, F.R. Meng, Y. Zhou, M. Zhu, M.Y. Shi, G.B. Sun, A node influence based label propagation algorithm for community detection in networks, Sci. World J. 2014 (2014) 627581, http://dx.doi.org/10.1155/2014/627581.

[30] K. Liu, J.B. Huang, H.L. Sun, M.J. Wan, Y.T. Qi, H. Li, Label propagation based evolutionary clustering for detecting overlapping and non-overlapping communities in dynamic networks, Knowl.-Based Syst. 89 (2015) 487–496, http://dx.doi.org/10.1016/j.knosys.2015.08.015.

[31] C. Tong, J.W. Niu, J.M. Wen, Z.Y. Xie, P. Fu, Weighted label propagation algorithm for overlapping community detection, in: Proceedings of IEEE International Conference on Communications, 2015, pp. 1–6, http://dx.doi.org/10.1109/ICC.2015.7248492.

[32] X.K. Zhang, J. Ren, C. Song, J. Jia, Q. Zhang, Label propagation algorithm for community detection based on node importance and label influence, Phys. Lett. A 381 (33) (2017) 2691–2698, http://dx.doi.org/10.1016/j.physleta.2017.06.018.

[33] S. Lim, J. Kim, J.G. Lee, BlackHole: Robust community detection inspired by graph drawing, in: Proceedings of IEEE International Conference on Data Engineering, 2016, pp. 25–36, http://dx.doi.org/10.1109/ICDE.2016.7498226.

[34] H. Gibson, J. Faith, P. Vickers, A survey of two-dimensional graph layout techniques for information visualisation, Inf. Vis. 12 (3–4) (2013) 324–357, http://dx.doi.org/10.1177/1473871612455749.

[35] T. Fruchterman, E. Reingold, Graph drawing by force-directed placement, Softw. - Pract. Exp. 21 (11) (1991) 1129–1164, http://dx.doi.org/10.1002/spe.4380211102.

[36] P. Gajdoš, T. Ježowicz, V. Uher, P. Dohnálek, A parallel Fruchterman–Reingold algorithm optimized for fast visualization of large graphs and swarms of data, Swarm Evol. Comput. 26 (2015) http://dx.doi.org/10.1016/j.swevo.2015.07.006.

[37] G.L. Xu, Z.Z. Song, Y. Wang, D.Y. Lin, J. Chen, T.Y. Mao, W.J. Xu, A graph layout framework combining t-distributed neighbor retrieval visualizer and energy models, IEEE Access 7 (2019) 27515–27525, http://dx.doi.org/10.1109/ACCESS.2019.2900358.

[38] O.H. Kwon, T. Crnovrsanin, K.L. Ma, What would a graph look like in this layout? A machine learning approach to large graph visualization, IEEE Trans. Vis. Comput. Graphics 24 (1) (2018) 478–488, http://dx.doi.org/10.1109/TVCG.2017.2743858.

[39] A. Noack, Modularity clustering is force-directed layout, Phys. Rev. E 79 (2) (2009) 026102, http://dx.doi.org/10.1103/physreve.79.026102.

[40] Z. Zhang, Social Psychology, second ed., People's Education Press, China, 2008.

[41] C. Cao, Z.Z. Chen, J. Caverlee, L.A. Tang, C. Luo, Z.C. Li, Behavior-based community detection: Application to host assessment in enterprise information networks, in: Proceedings of Conference on Information and Knowledge Management, 2018, pp. 1977–1985, http://dx.doi.org/10.1145/3269206.3272022.

[42] F.J. Weissing, Animal behaviour: Born leaders, Nature 474 (7351) (2011) 288–289, http://dx.doi.org/10.1038/474288a.

[43] C.C. Ioannou, M. Singh, I.D. Couzin, Potential leaders trade off goal-oriented and socially oriented behavior in mobile animal groups, Amer. Nat. 186 (2) (2015) 284–293, http://dx.doi.org/10.1086/681988.

[44] B. Collignon, J.L. Deneubourg, C. Detrain, Leader-based and self-organized communication: Modelling group-mass recruitment in ants, J. Theoret. Biol. 313 (2012) 79–86, http://dx.doi.org/10.1016/j.jtbi.2012.07.025.

[45] S. Lidgard, M.C. Carter, M.H. Dick, D.P. Gordon, A.N. Ostrovsky, Division of labor and recurrent evolution of polymorphisms in a group of colonial animals, Evol. Ecol. 26 (2) (2012) 233–257, http://dx.doi.org/10.1007/s10682-011-9513-7.

[46] A.M. Reynolds, Effective leadership in animal groups when no individual has pertinent information about resource locations: How interactions between leaders and followers can result in Levy walk movement patterns, Europhys. Lett. 102 (1) (2013) 18001, http://dx.doi.org/10.1209/0295-5075/102/18001.

[47] T. Wu, Y.X. Guo, L.T. Chen, Y.B. Liu, Integrated structure investigation in complex networks by label propagation, Physica A 448 (2016) 68–80, http://dx.doi.org/10.1016/j.physa.2015.12.073.

[48] S. Sandig, H.-U. Schnitzler, A. Denzinger, Echolocation behaviour of the big brown bat (*Eptesicus fuscus*) in an obstacle avoidance task of increasing difficulty, J. Exp. Biol. 217 (16) (2014) 2876–2884, http://dx.doi.org/10.1242/jeb.099614.

[49] M. Warnecke, B. Falk, C.F. Moss, Echolocation and flight behavior of the bat *Hipposideros armiger* terasensis in a structured corridor, J. Acoust. Soc. Am. 144 (2) (2018) 806–813, http://dx.doi.org/10.1121/1.5050525.

[50] H.L. Sun, J.B. Huang, Y.Q. Tian, Q.B. Song, H.L. Liu, Detecting overlapping communities in networks via dominant label propagation, Chin. Phys. B 24 (1) (2015) 018703, http://dx.doi.org/10.1088/1674-1056/24/1/018703.

[51] L. Lü, Y.C. Zhang, H.Y. Chi, T. Zhou, Leaders in social networks, the delicious case, PLoS One 6 (2011) http://dx.doi.org/10.1371/journal.pone.0021202.

[52] A. Lancichinetti, S. Fortunato, F. Radicchi, Benchmark graphs for testing community detection algorithms, Phys. Rev. E 78 (4) (2008) 046110, http://dx.doi.org/10.1103/PhysRevE.78.046110.

[53] W.W. Zachary, An information flow model for conflict and fission in small groups, J. Anthropol. Res. 33 (4) (1977) 452–473, http://dx.doi.org/10.1086/jar.33.4.3629752.

[54] D. Lusseau, The emergent properties of a dolphin social network, Proc. R. Soc. Lond. [Biol.] 270 (S2) (2003) S186, http://dx.doi.org/10.1098/rsbl.2003.0057.

[55] M.E.J. Newman, M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E 69 (2) (2004) 026113, http://dx.doi.org/10.1103/PhysRevE.69.026113.

[56] M. Girvan, M.E.J. Newman, Community structure in social and biological networks, Proc. Natl. Acad. Sci. USA 99 (12) (2002) 7821–7826, http://dx.doi.org/10.1073/pnas.122653799.

[57] M.E.J. Newman, Finding community structure in networks using the eigenvectors of matrices, Phys. Rev. E 74 (3) (2006) 036104, http://dx.doi.org/10.1103/PhysRevE.74.036104.

[58] D.E. Knuth, The Stanford Graph Base: A Platform for Combinatorial Computing, Addison-Wesley, 1993, pp. 41–43, http://dx.doi.org/10.1145/313559.313609.

[59] P.M. Gleiser, L. Danon, Community structure in jazz, Adv. Complex Syst. 6 (4) (2003) 565–573, http://dx.doi.org/10.1142/S0219525903001067.

[60] R. Guimera, L. Danon, A.D. Guilera, F. Giralt, A. Arenas, Self-similar community structure in a network of human interactions, Phys. Rev. E 68 (6) (2003) 065103, http://dx.doi.org/10.1103/PhysRevE.68.065103.

[61] D.J. Watts, S.H. Strogatz, Collective dynamics of small-world networks, Nature 393 (6684) (1998) 440–442, http://dx.doi.org/10.1038/30918.

[62] M. Boguna, R. Pastorsatorras, A. Diazguilera, A. Arenas, Models of social networks based on social distance attachment, Phys. Rev. E 70 (5) (2004) 056122, http://dx.doi.org/10.1103/PhysRevE.70.056122.

[63] M.E.J. Newman, The structure of scientific collaboration networks, Proc. Natl. Acad. Sci. USA 98 (2) (2001) 404–409, http://dx.doi.org/10.1073/pnas.98.2.404.

[64] J. Leskovec, K.J. Lang, A. Dasgupta, M.W. Mahoney, Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters, Internet Math. 6 (1) (2009) 29–123, http://dx.doi.org/10.1080/15427951.2009.10129177.

[65] J. Yang, J. Leskovec, Defining and evaluating network communities based on ground-truth, Knowl. Inf. Syst. 42 (1) (2012) 181–213, http://dx.doi.org/10.1007/s10115-013-0693-z.

[66] T. Chakraborty, A. Dalmia, A. Mukherjee, N. Ganguly, Metrics for community analysis: A survey, ACM Comput. Surv. 50 (4) (2017) 54:1–54:37, http://dx.doi.org/10.1145/3091106.

[67] A. Lancichinetti, S. Fortunato, J. Kertesz, Detecting the overlapping and hierarchical community structure in complex networks, New J. Phys. 11 (3) (2009) 033015, http://dx.doi.org/10.1088/1367-2630/11/3/033015.

[68] Y. Zhang, Y.G. Liu, J.T. Li, J.J. Zhu, C.H. Yang, W. Yang, W.C. Biao, WOCDA: A whale optimization based community detection algorithm, Physica A (2020) http://dx.doi.org/10.1016/j.physa.2019.122937.

[69] A.M. Fiscarelli, M.R. Brust, G. Danoy, P. Bouvry, Local memory boosts label propagation for community detection, Appl. Netw. Sci. 4 (2019) 95, http://dx.doi.org/10.1007/s41109-019-0210-8.

[70] C. Zhe, A. Sun, X. Xiao, Community detection on large complex attribute network, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2019, pp. 2041–2049, http://dx.doi.org/10.1145/3292500.3330721.

[71] A. Noack, Energy models for graph clustering, J. Graph Algorithms Appl. 11 (2) (2007) 453–480, http://dx.doi.org/10.7155/jgaa.00154.

[72] M. Jacomy, T. Venturini, S. Heumann, M. Bastian, ForceAtlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software, PLoS One 9 (6) (2014) e98679, http://dx.doi.org/10.1371/journal.pone.0098679.

[73] Y.C. Liu, Z.M. Li, H. Xiong, X.D. Gao, J.J. Wu, Understanding of internal clustering validation measures, in: Proceedings of the 2010 IEEE International Conference on Data Mining, 2010, pp. 911–916, http://dx.doi.org/10.1109/ICDM.2010.35.