J Syst Sci Complex (2019) 32: 737–758

A Link-Based Similarity for Improving Community Detection Based on Label Propagation Algorithm

BERAHMAND Kamal · BOUYER Asgarali

DOI: 10.1007/s11424-018-7270-1
Received: 17 August 2017 / Revised: 21 January 2018

Abstract Community structure is one of the most best-known properties of complex networks. Finding communities help us analyze networks from a mesoscopic viewpoints instead of microscopic or macroscopic one. It helps to understand behavior grouping. Various community detection algorithms have been proposed with some shortcomings in time and space complexity, accuracy, or stability. Label Propagation Algorithm (LPA) is a popular method used for finding communities in an almost-linear time-consuming process. However, its performance is not satisfactory in some metrics such as accuracy and stability. In this paper, a new modified version of LPA is proposed to improve the stability and accuracy of the LPA by defining two concepts -nodes and link strength based on semi-local similarity-, while preserving its simplicity. In the proposed method a new initial node selection strategy, namely the tiebreak strategy, updating order and rule update are presented to solve the random behavior problem of original LPA. The proposed algorithm is evaluated on artificial and real networks. The experiments show that the proposed algorithm is close to linear time complexity with better accuracy than the original LPA and other compared methods. Furthermore, the proposed algorithm has the robustness and stability advantages while the original LPA does not have these features.

Keywords Community detection, complex networks, LPA, similarity measure.

1 Introduction

A complex system is a special real-world system, i.e., a set of interacting elements relatively isolated from their environment, and possessing some emerging properties[1]. By exploring the real world, we can find many complex systems such as the social network[2], the Internet[3], WWW[4], transportation systems[5], protein-protein interactions[6, 7], brain networks[8], and finical and economic systems[9]. Such complex systems can be modeled as complex networks in which entities are represented as nodes and the relationship among entities are defined as links. Complex networks are widely used for modeling real-world systems in very different areas.

BERAHMAND Kamal · BOUYER Asgarali (Corresponding author)
Department of Information Technology and Communications, Azarbaijan Shahid Madani University, Tabriz, Iran. Email: berahmand@azaruniv.ac.ir; a.bouyer@azaruniv.ac.ir.
◇ *This paper was recommended for publication by Editor LÜ Jinhu.*

They have non-trivial topological features, due to the particular features of the complex system. Therefore, this network-or graph- differs from random and regular networks. For instance, complex networks have small diameters which conclude the small world phenomenon. The next features are the high clustering coefficient, heavy tail in the degree distribution (power-law), robustness, assortativity or disassortativity among nodes, community structure, etc.[1, 10, 11].

Community structure is an important property for analyzing and visualizing given networks from mesoscopic viewpoints. Community structure can be identified when the distribution of links among nodes is too homogeneous[12]. Over the recent decade, there have been many investigations in defining and finding the community structure of complex networks. Community structures, also called clusters or modules, are groups of nodes which have dense connections within a community and sparse connections among communities. In social networks, the community is instantly defined as the social group of users sharing common interests in the network[13]. In the World Wide Web complex network, a community is a group of websites dealing with the same topics. In the brain network, communities probably correspond to different local body controllers[14]. In metabolic networks, communities may be related to functional modules such as cycles and pathways or a group of proteins functioning in a similar way within a cell in the protein-protein interaction networks[15].

In recent years, many studies have been conducted on the subject of discovering communities. These algorithms try to maximize links within communities while minimizing links among communities[12]. However, most of these methods are inapplicable due to their low degree of accuracy or finding communities in large time and space complexity. For example, because of the exponential growth of real-world graphs such as social networks and biological networks, we need to use a community detection algorithm with an almost-linear time and space complexity. One of the most popular almost-linear methods is Label Propagation Algorithm (LPA)[16]. LPA has low time complexity and is well-adapted in the large-scale community. The main goal of LPA is to assign a unique label to each node, update the label of the node with the most seen label among its neighbors, and finally to discover community-based on the last updated stage. Despite the simplicity and time efficiency of LPA, it has some deficiencies, namely its randomly selected initial node, random updating sequence and random tiebreaking[17–19]. In recent studies, many modifications of LPA have been proposed to improve its accuracy and robustness. However, these methods suffer from two main weaknesses: 1) The dependence of LPA on free parameters and 2) the use of global information structure. We have discussed these weaknesses in many studied LPA-based methods in Section 2.

In this paper, a new LPA-based algorithm is proposed to improve the accuracy and stability of the original LPA. A strength value has been defined for nodes and links based on semi-local similarity. This proposed algorithm eliminates two random steps of the original LPA and uses node strength for the updating order and link strength for rule updating. The proposed algorithm is then tested on artificial and real networks. The experiments show that the proposed algorithm is close to linear time complexity and can detect communities with better quality and stability.

The structure of this paper is as follows. Section 2 discusses recent studies on community detection topics. In Section 3, the modified algorithm is proposed based on the label propagation concept. Section 4, real and artificial data sets have been introduced as well as evaluation criteria for real and artificial data sets. The proposed algorithm is tested on artificial and real datasets to verify its efficiency and accuracy in Section 5. Finally, Section 6 concludes the paper and proposes future directions.

2 Related Work

Hierarchical clustering is one of the oldest and most popular methods for finding communities (clusters) in complex network analysis. There are some methods based on hierarchical clustering algorithms such as GN[15], edge clustering coefficient[20], and Information centrality[21].

Modularity-based methods try to detect communities using modularity metric. These methods suppose a high modularity value for a well-separated community. Newman proposed a fast algorithm for detecting community structure based on the modularity maximization[22]. This method has a time complexity of $O((m+n)n)$ on an arbitrary network and $O(n^2)$ for a sparse network. The Fastgreedy method is a faster version of the previous algorithm, which uses more efficient data structures (Maxheap tree) for updating[23]. The execution time of this method is $O(md \log n)$, where d is the depth of the dendrogram describing the network's community structure. Total time complexity is totally $O(n \log^2 n)$ on a sparse network. BGLL algorithm is a greedy method based on local information for optimizing modularity[24]. This method consists of two steps. First, it finds small or initial communities by optimizing of modularity using local information. Then, it builds a new network whose nodes are the communities found during the first phase. These steps are iteratively repeated until no more changes are needed and a maximum of modularity is attained. The time complexity of BGLL method is $O(n \log n)$, where n is the number of nodes in the network. However, all modularity-based algorithms suffer from a resolution limit beyond which no smaller community can be detected. It has been proved that modularity Q is not a scale-invariant measure, and consequently, it is unable to detect communities smaller than a certain size depending on its maximization[25].

In random walk methods, each node initially contains a walker. Then each walker will randomly choose a neighbor of the node it currently stands on to localize. The idea behind Random Walk is that the walk tends to be trapped in dense parts of a network to Communities[26]. Markov Clustering (MCL) algorithm uses an initial stochastic matrix M for clustering by iteratively repeating two main operators-expansion and inflation-until convergence[27]. WalkTrap approach defines a similarity measure based on random walks and applies a hierarchical agglomerative clustering fashion for discovering an adequate number of communities using modularity measure[26]. Infomap is the most popular algorithm among random walk methods which is a flow-based community detection by combining information-theoretic techniques and random walks[28]. Infomap method inherently turns community detection problem into a coding problem. Using Huffman coding, the problem of finding communities is solved on two levels: One level to distinguish communities in the network and the other to distinguish nodes in a community. Nodes in different communities can reuse the same code. The discovered partitions of networks minimize the description length of an infinite random walk process.

Label propagation algorithm (LPA) is a popular and fast method for community detection proposed by Nandini, et al.[16]. Initially, for each node in the network, a unique label is assigned. At the next step, each node updates its label with the most frequent label among its neighbors. When some labels of neighbors are equally frequent, it randomly chooses from the most frequent labels. This label propagation process is repeated until the nodes with the same label are grouped into one community. The main advantages of LPA are to have an almost-linear time complexity, to use the local information, to not depend on free parameters and objective function, and to be simple in simple implementation[29]. However, this algorithm has some weaknesses such as instability, low quality, and formation of monster communities due to its random behavior in initial node selection and random updating the label of a node randomly in the tiebreak state. Recent investigations show that the various modifications of LPA have been implemented in order to improve its stability and robustness. However, these methods have focused either on initial node selection or rule updating. Leung, et al. proposed hop attenuation and preferential linkage to break ties and prevent monster communities on large web networks[17]. This technique avoids the label from spreading too far from its origin. Šubelj and Bajec proposed a dynamic hop attenuation and hybrid label propagation method called diffusion and propagation algorithm (DPA)[19]. This approach retains the dynamics of label propagation and still prevents the emergence of a monster community. Barber and Clark proposed a modularity-specialized LPA (LPAm) which uses an objective function with the aim of reaching a high modularity value[30]. However, it tends to fall into poor local maxima in the modularity space[18]. To solve this problem, Liu and Tsuyoshi presented an advanced modularity-specialized LPA (LPAm+) which combines multi-step greedy agglomerative algorithm (MSG) with LPAm. Since LPAm and LPAm+ require the use of modularity metric, they inherently suffer from the weakness of modularity-based methods. Labelrank is another extension of LPA which is an extended version of Markov Cluster Algorithm[31]. It stores, propagates and ranks the labels in each node. It performs four customized tasks called propagation, inflation, cut off and conditional update to stabilize the dynamic propagation. LPA-CNP method introduces a weighted coherent neighborhood propinquity (weighted-CNP) measure to find stable communities in LPA[32]. Liang, et al. proposed a new development on LPA by applying consensus clustering technology (LPAcw)[33].

In LPAcw (consensus weight) algorithm, the weight of the edge is the proportion of the times number which two nodes of the edge are assigned in the same community to the total consensus time. CK-LPA is a new developed LPA based on community kernel detection. It initially tries to detect community kernel by selecting seed nodes in degree discount manner and expands seed nodes by selecting node maximization degree[34]. Then, CK-LPA allocates a weight for each node. It can be concluded conclude that when the nodes are far from the community kernel, the weight will be low. Therefore, update rule is done by the label with the maximum summation of weight values is chosen as the new label. Sun, et al. proposed CenLP algorithm which extends initial node selecting and updating of LPA[29]. For each node, it assigns a local density value ρ and the similarity δ with the assumption that the cores in communities favor a higher local density and lower similarity. Therefore, the weight of each node is computed using relation $\gamma = \frac{\rho}{\delta}$. The updating order can be obtained in an ascending order of for nodes. For each node, a large value of γ reveals that the node has a high potential for being the core in a networks. For node preference, a node prefers to follow its neighbor whose local density is higher than itself. Then, among these higher local density nodes, it selects a node with maximum similarity.

3 The Proposed Efficient Community Detection Algorithm

In this section, we explain our proposed algorithm as a new development of LPA. We have used two new concepts, node and link strengths, and it is necessary to define these concepts at below.

3.1 Definitions

Generally, a network can be described by a graph $G(N, E, W)$, where N is the set of vertices, E is the set of edges, and $W(E)$ is the weight of the edge E.

Definition 3.1 (link strength) It is defined based on the similarity between its pair nodes using Equation (1)[35]. The neighborhoods (α) of N_i is defined as neighborhoods (α) $(N_i, \alpha) = \{N_i \in N \mid \text{dist}(N_i, N_j) \leqslant \alpha\}$ where α shows the allowable depth of common neighbors with subject $\alpha > 0$, and dist (N_i, N_j) is the shortest path length between N_i and N_j. In this definition α is set to 2, because the effect of $\alpha > 2$ is negligible on the link strength and A_{ij} adjacent matrix is negligible.

$$\text{sim}(N_i, N_j \mid \alpha) = A_{ij} \frac{|\text{neighborhoods}(\alpha)(N_i, \alpha) \cap \text{neighborhoods}(\alpha)(N_j, \alpha)|}{|\text{neighborhoods}(\alpha)(N_i, \alpha) \cup \text{neighborhoods}(\alpha)(N_j, \alpha)|}. \tag{1}$$

Equation (1) shows a similarity metric to quantify this fact that the nodes in the same community tend to have more common nodes in their neighborhoods. In fact, this formulation is an extended version of Jaccard's similarity, which in addition to the first-level of neighbors, consider the second-level of neighbors to be more precise.

Definition 3.2 (node strength) In a given weighted network $G(N, E, W)$, extending the definition of node degree

$$K(u) = \sum_{N \in \Gamma(u) \setminus \{u\}} (\text{sim}(u, v)). \tag{2}$$

$\Gamma(u)$ shows the structure neighborhood for a node $u \in V$. The structure neighborhood for the node u is the set $\Gamma(u)$ containing u and its adjacent nodes which share a common incident edge with $u : \Gamma(u) = \{v \in V \mid \{u, v\} \in E\} \cup \{u\}$.

Equation (2) shows that in unweighted graphs, the degree centrality of each node is calculated based on the number of its links, but in weighted graphs the total weight of the links of each node is calculated as node strength. In other words, this equation measures the strength of nodes in terms of the total weight of their connections.

Definition 3.3 (node preference) In a given weighted network, the preference of a node N is defined as follows:

$$\text{Node Preference}(N) = \Big\{ u \in V \mid \operatorname*{argmax}_{u \in \Gamma_{1(v)}} \{\text{sim}\{v, u\}\} \Big\}. \tag{3}$$

Therefore, for node preference, it prefers to follow the neighbor whose local density is higher than the node itself and similarity with the node is highest among neighbors.

3.2 Label Propagation Algorithm

In this section, we bring an example for original LPA algorithm to show its instability and other disadvantages.

Example 3.4 Take the simple network with 18 nodes and 31 links shown in Figure 1. There are original two communities in this network (painted in gray and blue colors respectively):

$$\text{Gray community}: (N_1, N_2, N_3, N_4, N_5, N_6, N_7, N_8),$$
$$\text{Blue community}: (N_9, N_{10}, N_{11}, N_{12}, N_{13}, N_{14}, N_{15}, N_{16}, N_{17}, N_{18}).$$

Feeding LPA with this network, we may obtain just one community (unsuccessful case). Let's consider an unsuccessful updating rule in Figure 2. As mentioned above, in the original LPA, the result of network partitioning is sensitive to the updating order of node labels during label propagation. Assume that nodes in the left community (nodes 1–8) are already assigned with label 3 after some steps, while the nodes in the right community are labeled with the number of 9–18, respectively.

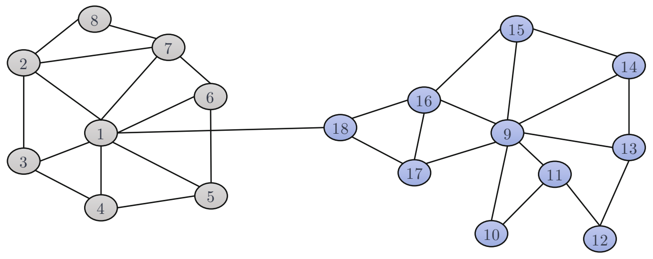

Figure 1 A simple network with 18 nodes, 31 links, and 2 communities

As shown in Figure 2, if we select the updating order for the blue community in the order of 18, 16, 17, 9, 15, 14, 13, 12, 11, and 10 respectively, node 18 is updated first. It may choose label of nodes 1, 16, or 17 with the same probability. In this case, it uses tiebreak strategy to choose a label from its neighbors randomly. If it chooses the label 3 of node 1, its label is updated with label 3. In Step 2 of Figure 2, this condition is repeated for node 16. If node 16 similarly uses a tiebreak strategy, it may update its label with label 3 of node 16. Therefore, in Step 3, node 17 is eventually labeled with 3. At the end, the outcome corresponds to an inappropriate solution, where all nodes in the network are categorized under the same community. However, it is possible to reach a true community finding in this order if node 18's label is firstly updated its label with the label of node 17. Then, nodes 16, 9, 15, 14, 13 will be updated with label 17 and node 12 in a tiebreak strategy may be updated with label 17. Likewise, node 11 can be updated with label 17 and node 10 is consequently updated with label 17. At the end, all nodes in the blue community are labeled with 17 and the gray community is labeled with 3. On the other hand, let us consider another updating order as order 13, 14, 12, 11, 15, 16, 17, 18, 9, and 10, respectively. As shown in Figure 3, suppose that the node 13 chooses the label of node 9 in a tiebreak strategy in Step 1. Therefore, in Step 2, node 14 is also updated with label 9. In Step 3, node 12 may also choose label 9 in tiebreak state. Step 4 shows that the node 11 is eventually updated with label 9. At the end of Step 9, the outcome will correspond to the true community partitioning of the network.

This example reveals that the LPA is heavily dependent on the updating order and tiebreak strategy. In original LPA, a label of a node is randomly updated with one of the labels of neighbors that make instable decisions in finding community structure. Correspondingly, another weakness of original LPA is the production of monster communities, shown in Step 9 of Figure 2.

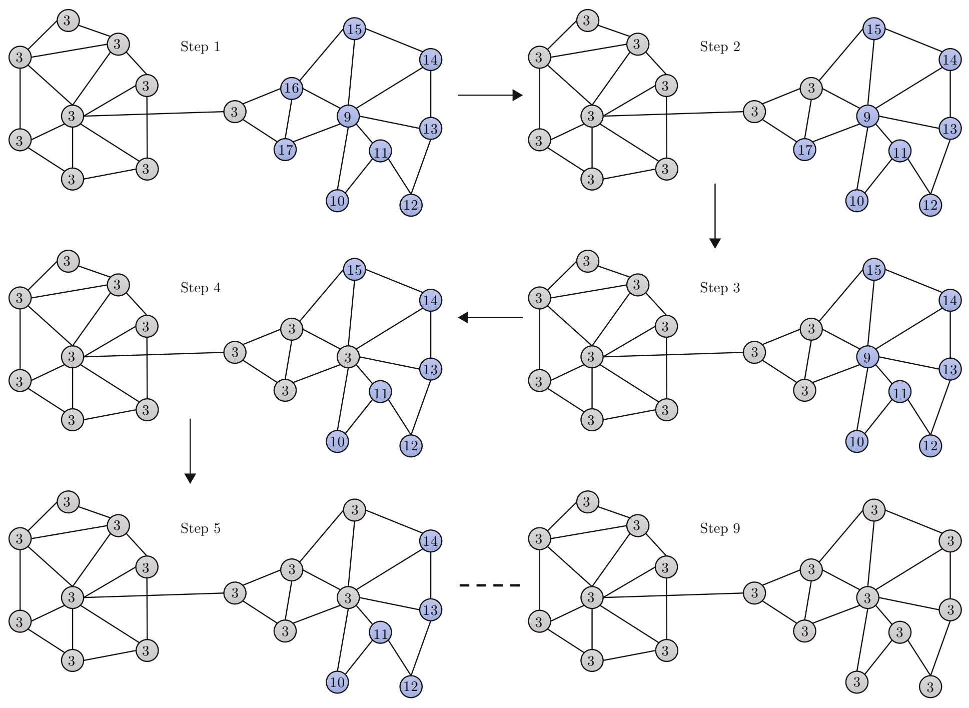

Figure 2 An example of unsuccessful updating order (18, 16, 17, 9, 15, 14, 13, 12, 11, 14)

3.3 The Proposed LPA-Based Algorithm

As mentioned in Subsection 3.2, LPA has some problems that make it inefficient for many networks. In this paper, we have proposed an optimized version of LPA that solves problems of the original LPA and other LPA-based algorithms such as stability, monster communities, and low performance.

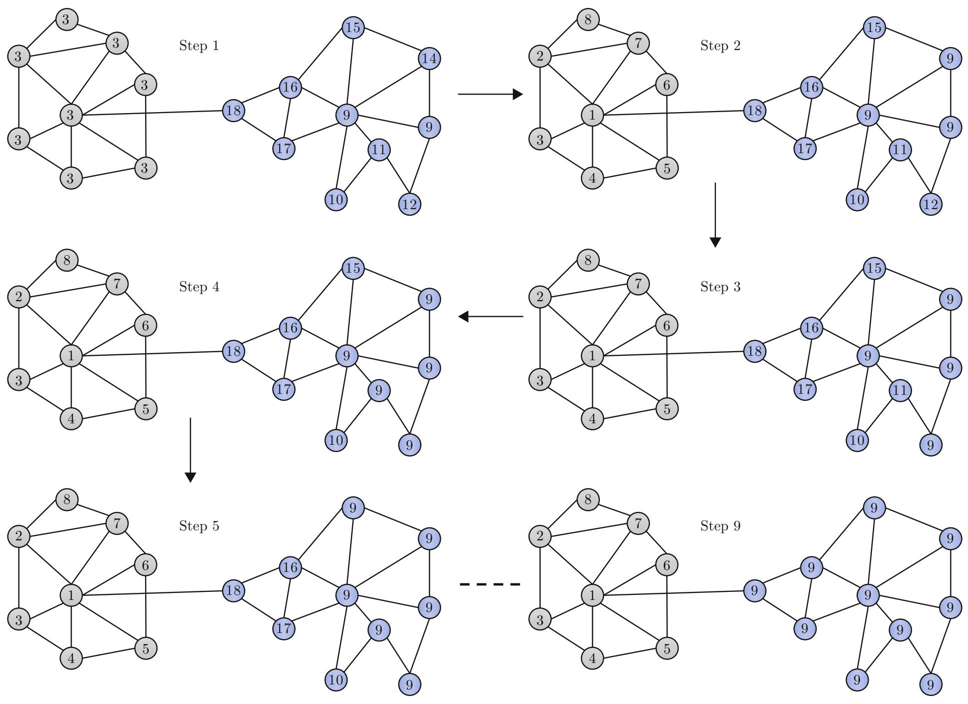

Figure 3 An example of successful updating order

In LPA, all nodes have equal importance of selecting and propagating of its label to other nodes due to using a random method for initial and tiebreaks selections. In fact, it does not consider the node's characteristics in starting and updating steps. So, it leads to propagating labels among different communities randomly that consequently affect the accuracy of LPA. According to [36], the nodes of a complex network are divided into four groups: 1) Core, 2) Hub, 3) Bridge, and 4) Periphery. For detecting communities using the LPA algorithm, the hub and bridge nodes have negative effects because of propagating an invalid label to other communities. On the other side, the core nodes are the most important elements due to their positive effect on label propagation inside the community. The core nodes have high degree centrality, eigenvector centrality, and ago betweenness. Therefore, core nodes are suitable nodes to be used in start and update phases. However, the core nodes must be detected using a local or semi-local method to avoid further time complexity. In other words, the popularity of original LPA algorithm is its linear time complexity ($O(m)$) on account of using a local method for label propagation. Therefore, we need to preserve the time complexity of LPA in the proposed algorithm. Nevertheless, nodes inside a community have higher structural similarity than nodes in other communities[37]. Thus, if the nodes' importance is used for selecting a node in the initial step instead of a random selection of LPA, this algorithm will be stable and will prevent the formation of monster communities. A pseudo-code for our proposed algorithm is shown in "Algorithm 1". In general, the steps of the proposed algorithm are denoted as follows:

Step 1 Initialization – At the starting phase, a unique label is assigned to each node in the network. Each label reveals the community of the node. We need to remove random node selection of original LPA in our proposed algorithm. Hence, Equation (1) is used for computing the link strength. Next, the strength of a node is calculated using Equation (2). Then, these nodes are sorted in descending order based on their computed node strength. Finally, the node with the highest strength is selected as the initial node.

Step 2 Label propagation – In label updating for nodes, we use the node's preference according to Equation (3). All nodes in the network are iteratively updated in descending order of node's strength value. In each iteration, the label of each node is determined by the label of its neighbor with the highest link strength among neighboring nodes. If the tie-breaks state is happening, the algorithm will select an edge whose target node has a higher strength than other neighbors. In other words, if there are several edges with equal but highest weight, a node label is chosen based on maximum node's strength among neighbors.

Step 3 Nodes with the same label are assigned to the same community.

Algorithm 1: The proposed LPA-based community detection algorithm

```
   Input  : network $G = (N, E)$
   Output: Community structures $C = \{C_1, C_2, \cdots, C_K\}$
1  Assign a unique label to each node in the network and Set $t = 1$.
2  Calculate the link strength and node strength using Equations (1) and (2), respectively
3  While the label of nodes change or $t <$ max iteration do:
4      Arrange node in ascending order of node strength and put the results on the vector
       $X$.
5      For each node $v_i \in X$ vector, update its labels according to Equation (3).
6      If tiebreak state happens, choose the neighbor node who has higher node's strength.
7      $t = t + 1$.
8  End While
9  Construct communities based on the similar label.
10 Return community structures.
```

Example 3.5 (Testing the proposed algorithm on the network of Figure 1) At the start, the link strength is calculated using semi-local centrality (Equation (1)). The node's strength is then computed for each node based on link strength by Equation (2). As it can be seen in Figure 4, all nodes are uniquely labeled in the beginning. Then, the node 1 is selected for updating its label because of its higher node's strength value. It takes the node's label from its neighbors which have a higher edge's strength. However, in some nodes, several edges have same strength value among the target node and its neighbors (tiebreak state). In this case, it selects the label of a neighbor which has a higher node's strength value. For example, the link strength between node 1 with nodes 7, 2, and 5 is the same. Therefore, it updates its label with the label of node 7 due to its higher node's strength value. After node 1, node 9 is updated. It updates its label with the label of node 16 because of higher link strength between nodes 9 and 16 than other neighbors. Likewise, all other nodes also update their labels according to their strength value, likewise. At the end of iteration 1, nodes 1 to 8 are gathered in the same community because they receive the label of node 7. And, the nodes 9 to 18 are assembled in another community because their label is updated with the label of node 16. This process is repeated in iterations 2 and 3 until the global optimum is reached. In most cases, however, for small networks, iterations 1 and 2 are enough in most of the time.

Table 1 Iterations of our algorithm based on Example 3.5

Node	Iteration 1			Iteration 2			Iteration 3		
	Order updating	Current label	New label	Order updating	Current label	New label	Order updating	Current label	New label
1	1	1	7	1	7	7	1	7	7
2	9	9	16	9	16	16	9	16	16
3	7	7	7	7	7	7	7	7	7
4	2	2	7	2	7	7	2	7	7
5	16	16	16	16	16	16	16	16	16
6	6	6	7	6	7	7	6	7	7
7	3	3	7	3	7	7	3	7	7
8	4	4	7	4	7	7	4	7	7
9	5	5	7	5	7	7	5	7	7
10	15	15	16	15	16	16	15	16	16
11	14	14	16	14	16	16	14	16	16
12	17	17	16	17	16	16	17	16	16
13	13	13	16	13	16	16	13	16	16
14	11	11	16	11	16	16	11	16	16
15	18	18	16	18	16	16	18	16	16
16	8	8	7	8	7	7	8	7	7
17	10	10	16	10	16	16	10	16	16
18	12	12	16	12	16	16	12	16	16

3.4 Computational Complexity of the Proposed Algorithm

We assume that the number of nodes in a social network is n, and the number of edges or link is m. For analyzing time complexity, we process it in several isolated steps. Each step individually runs on a different time complexity. The nodes are initially labeled in time $O(n)$. The next step is calculating node and link strength. For calculating link strength, we need to have access to neighbors and neighbor of neighbors. Therefore, its time complexity for all nodes is $O(nk^2)$ which is equal to $O(m)$ because complex networks often have a massive graph with a very small number of neighbors. Therefore, the number of neighbors does not have an effect on total time complexity. The node strength is also computed in $O(nk)$ due to the fact that only the neighbors are taken into consideration. The third step is, ranking nodes based on node strength that has time complexity $O(n)$ (due to the possibility of using radix and bucket sorting algorithm in a linear time). The next step is the label propagation process. The time complexity of label update according to weight link neighbor is $O(nk)$ which is equal to $O(m)$. Finally, the time complexity of assigning the nodes with the same label to their own community is $O(2n)$. Since scale-free networks have sparsity property, the number of links is nearly equal to the number of nodes. Therefore, The time complexity of $O(m) \approx O(n)$. Consequently, the total time complexity is

$$T(n) = O(nk^2 + nk + 4n + m) \approx O(m).$$

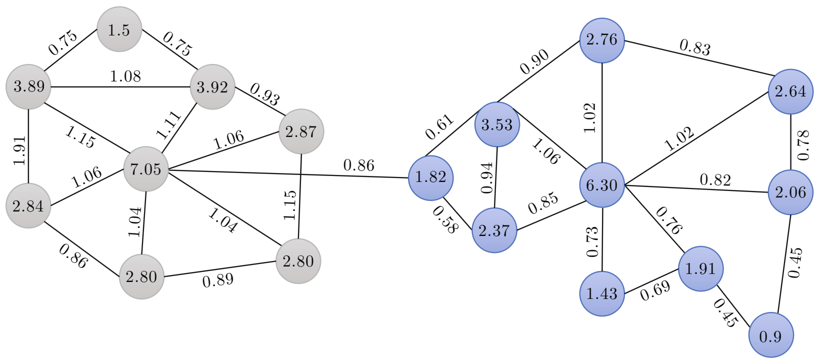

Figure 4 An example of the label propagation process by our proposed algorithm on the network of Figure 2

4 Experiments

In this section, we evaluate the efficiency of our proposed algorithm with other state-of-the-art community detection algorithms such as CNM[23], Infomap[28], BGLL[24], and original LPA[16]. CNM and BGLL algorithms are selected for comparison due to their higher accuracy among modularity-based methods. We also consider LPAm+and LPA-CNP as improve method LAP proposing recently. Our algorithm is programmed in C++ with the GCC compiler. All experiments were conducted on a PC with an Intel core i5 CPU (2.8 GHz) and 6.0 GB of memory.

4.1 Datasets

We have conducted our experiments on two types of datasets, computer-generated network (LFR) datasets and ten real network datasets. Girvan and Newman (GN)[15] benchmark networks are not used in this evaluation because all the node degrees follow a Poisson distribution in GN benchmarks that make them non-real networks. Therefore, LFR networks are adequate for this experiment.

4.1.1 Artificial Datasets

LFR Benchmark network: LFR Benchmark was proposed by Lancichinetti, et al.[38]. It represents the realistic properties similar to real-world networks because of the controlled power-law node degree distribution, and community size distribution. In LFR, we can produce different datasets by varying the parameters of the network. Some important parameters are detailed are presented in Table 2.

Table 2 The main parameters of LFR datasets

Parameter name	Description
N	Number of nodes
K	Average degree
$\max(k)$	The maximum degree
γ	The exponent for the degree distribution
β	The exponent for community size distribution
$\min(c)$	The minimum community size
$\max(c)$	The maximum community size
μ	Mixing parameter

μ is the mixing parameter which shows the fraction of the link for each node connected to other communities using Equation (4):

$$\mu(n \in C) = \frac{\sum_{\forall C' \in P, C'C} \mathrm{com}(n, C')}{d(n)}, \tag{4}$$

where $d(n)$ is the number of edges of node n, $\mathrm{com}(n, C')$ is the compatibility of node n to community C' (or the number of direct neighbors of node n in C'). A larger value of μ indicates a less clear community structure of the network.

We generate six groups of LFR benchmark networks. Each group contains eight networks with μ ranging from 0.1 to 0.8 and common parameters N, K, $\min(c)$, and $\max(c)$. In other words, these parameters are the same in networks of each group but different in networks of other groups. The other parameters are set to default values. The details are shown in Table 3.

Table 3 The parameters of six groups of LFR networks

Networks	N	K	$\max(k)$	$\min(c)$	$\max(c)$	μ
$N1$	5000	15	40	10	40	0.1–0.8
$N2$	5000	15	40	40	80	0.1–0.8
$N3$	10000	20	50	20	50	0.1–0.8
$N4$	10000	20	50	50	100	0.1–0.8
$N5$	50000	40	100	50	100	0.1–0.8
$N6$	50000	40	100	100	200	0.1–0.8

4.1.2 Real Datasets

The next experiment is based on ten standard real networks. Detailed information of each network is shown in Table 4. In these networks, N is the number of vertices, E is the number of edges and C is the number of communities. The parameter C is unknown for networks $E5$ to $E10$.

Table 4 The parameters of six groups of LFR networks

Network ID	Network Name	N	E	C	Refrences
$E1$	Karate Club of Zachary	34	78	2	[39]
$E2$	Dolphins	62	159	2	[40]
$E3$	American College Football network	115	613	12	[15]
$E4$	Political Books network	105	441	3	[41]
$E5$	Jazz	198	2742	—	[42]
$E6$	C.Elegans	453	2032	—	[43]
$E7$	Email	1133	5451	—	[44]
$E8$	Netscience	1589	2742	—	[45]
$E9$	PowerGrid	4941	6494	—	[46]
$E10$	Internet	22936	48436	—	[48]

Karate Club of Zachary ($E1$) The Karate Club of Zachary is a social network consisting of friendships among 34 people with 78 relationships in a karate club at a US university, as described by Wayne Zachary in 1977. At some point, the club president had a dispute with an instructor that led to the split of the club into two separate groups, supporting the instructor and the president, respectively. Figure 7 represents the original karate club network.

Dolphins ($E2$) This dataset is the complex network of bottlenose dolphins reported by Lusseau. It is an undirected social network with 62 nodes and 159 edges; each node denotes a dolphin and each edge between two dolphins reveals the frequent contact between two dolphins, living in Doubtful Sound Gulf, New Zealand. The dolphin dataset is originally divided into two communities. Figure 9 Represents the original dolphin network.

American College Football network ($E3$) This network consists of 11 different regular conferences of participating teams, except for 8 independent teams. Sometimes, these 8 teams are considered as the 12th conference. There are 115 Division I-A teams (nodes) that play 613 games (edges) during the regular fall season of 2000. Each node denotes a football team and an edge between two nodes shows a game played by these teams. Each team approximately plays a total of 7 intra- and 4 inter-conference games in the season.

Political Books Network ($E4$) This network, compiled by Krebs, consists of the political books about US politics in 2004 presidential election that were published around 2004 and sold by Amazon. A node represents a political book and a link between two nodes represents frequent co-purchasing of books by the same buyers.

Jazz ($E5$) This dataset is a collaboration network among Jazz musicians that was created in 2003. Each node is a Jazz musician and an edge indicates the relationship between two musicians. The relationship is established between two musicians while they were performing together in a band.

C. Elegant ($E6$) This network shows metabolism interactions for a typical worm, called C.Elegant. This dataset has 453 nodes (metabolites) and 2032 edges. Each edge represents a metabolic reaction between two nodes.

Email ($E7$) This dataset consists of 1669 users including faculty members, researchers, technicians, managers, administrators, and graduate students. Each user has an email in the email dataset. In this network, each email address is defined as a node and email communication between two nodes is considered as an edge. Out of the total 1669 nodes, 1133 belong to the giant component.

Netscience ($E8$) Co-authorship in science network or netscience is a network with 1589 co-authors or scientists that are skilled in network theory and practice. It was compiled by M. Newman in May 2006. Each node is a scientist and an edge represents the relationship among scientists. This network consists of 1589 node and 2742 edges.

PowerGrid ($E9$) It is an undirected and unweighted network. This network illustrates the topology of the Western States Power Grid of the United States. The power grid dataset includes 4,941 nodes and 6,594 edges. A node is a generator, a transformation, or a substation and an edge represents a power supply line.

Internet ($E10$) A symmetrized snapshot of the structure of the Internet at the level of autonomous systems, reconstructed from BGP tables posted by the University of Oregon Route Views. This snapshot was created by Mark Newman using the available data in July 22, 2006.

4.2 Evaluation Criteria

In this section, we describe Normalized Mutual Information (NMI) and modularity (Q) criteria as the evaluation metrics, currently popular in measuring the performance of network clustering algorithms. Normalized Mutual information: The NMI is computed using Equation (5)[47]:

$$\mathrm{NMI}(A, B) = \frac{2I(A, B)}{H(A) + H(B)}, \tag{5}$$

where A and B denote two partitions of the network, $H(A)$ and $H(B)$ are the entropies of each partition and $I(A, B)$, ranging from 0 to 1, represents the mutual information between A and B. If NMI=1, it means that partition A is identical to partition B. If partition A is totally independent of partition B, for example when the entire network is found as just one community, then $I(A, B) = 0$.

Modularity (Q) It is a criterion which rates the quality of a network clustering. Modularity metric is computed by Equation (6). If the Q is closer to 1 it reveals high-quality clustering (communities). Modularity compares the actual number of intra community edges to the expected number of edges in a random graph with the same degree distribution. The modularity

(Q) is defined as:

$$Q = \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{d_i d_j}{2m} \right) \times \delta(c_i, c_j), \tag{6}$$

where m represents the number of edges, A_{ij} is an adjacency matrix, where $A_{ij} = 1$ if node i is linked to node j. Otherwise, $A_{ij} = 0$. $\delta(c_i, c_j)$ is a piecewise function is defined as $c_i = c_j$, then $\delta(c_i, c_j) = 1$, else $\delta(c_i, c_j) = 0$.

5 Experimental Results and Discussion

At first, the artificial datasets are tested by mentioned five algorithms. The obtained results are illustrated in Figures 6 to 11. These six figures show the NMI of the proposed algorithm and other compared methods on six groups of LFR benchmark networks (N1∼N6). The abscissa represents the parameter μ from 0.1 to 0.8. Next, we have evaluated our algorithm on real well-known networks.

5.1 Analysis of the Synthetic Networks

Since our proposed algorithm is a developed version of LPA, we compare it with the original LPA on the datasets for a reliable validation of stability and accuracy based on NMI metric. In these experiments, we have implemented LPA algorithm in 100 runs on datasets N1 to N6 because of its random nature and sensitivity of initial inputs. The fluctuation of LPA is shown in these figures. As mentioned above, the proposed algorithm tries to avoid a random decision; as a result, it does not have any fluctuations. Therefore, it is more and more stable than LPA-based algorithms such as CNM, BGLL, LPA-CNP and LPAm+. However, our algorithm is also run 100 times to validate the stability. The obtained results have also verified our algorithm's claimed stability.

Figure 5 shows the results of six algorithms in LFR networks using different parameters listed in Table 1. These figures illustrate the comparison of NMI metric for community detection of our algorithm and other five algorithms. These experiments prove that our algorithm significantly outperforms other algorithms based on NMI metric. All the methods, except for Fastgreedy algorithm, have better performance when the mixing parameter's value (μ) is small.

When μ is increased, the network is more complex and it becomes more difficult to reveal the community. For example, the performance of CNM algorithm in $\mu \geqslant 0.7$, BGLL algorithm in $\mu \geqslant 0.6$, LPA-CNP algorithm $\mu \geqslant 0.4$ LPAm+ algorithm in $\mu \geqslant 0.3$, LPA algorithm in $\mu \geqslant 0.5$ drops for all datasets N1 to N6. Needless to say, our algorithm's performance is better than compared methods in all LFRs. In LFR N1, our algorithm's performance is similar to BGLL in $\mu = 0.6$. However, for other values of μ, our algorithm is better than compared methods. In LFR N2, our algorithm's performance is similar to BGLL, LPA. For other values, the performance of our algorithm is better than other algorithms. In LFR N3 and N4, it can be seen that the NMI value for our algorithm is higher than other algorithms for all μ's possible values. As shown in figures of LFR N5 and N6, the performance of our algorithm is similar to BGLL. Needless to say, our proposed method also has a better performance than other algorithms for all μ values. Consequently, our algorithm is not weaker than any compared methods. It significantly outperforms the compared method in all LFR datasets.

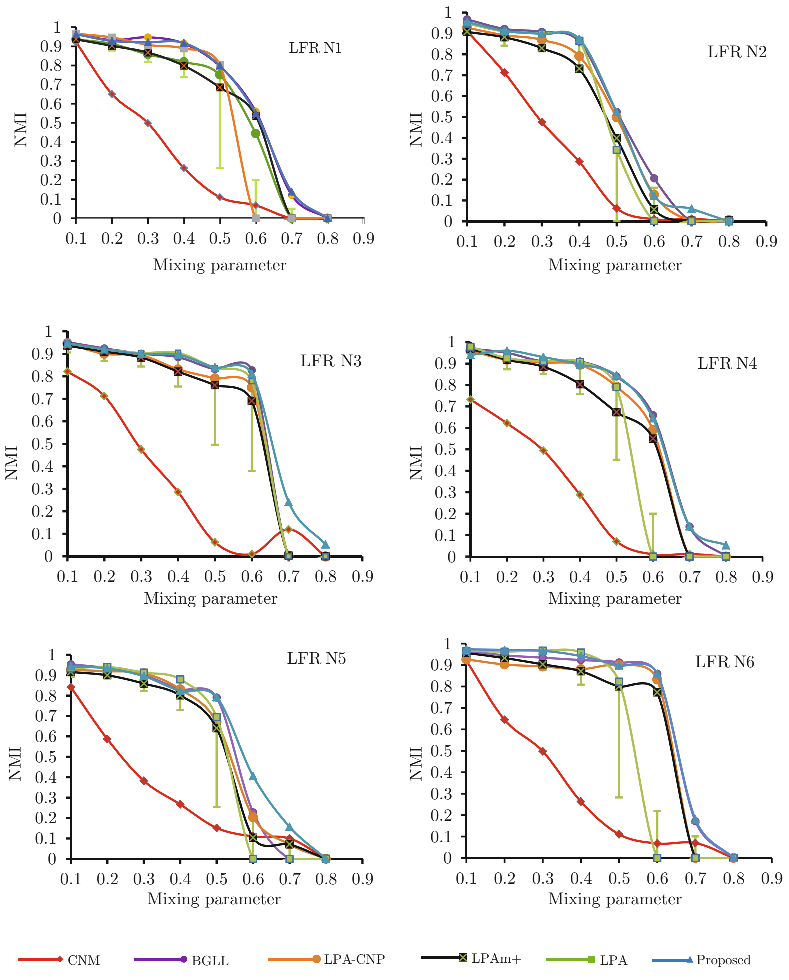

Figure 5 The experimental results on networks LFR N1, LFR N2, LFR N3, LFR N4, LFR N5, and LFR N6

5.2 Analysis on Real-World Networks

In this section, the performance of our algorithm is compared with the other algorithms in real-world networks using the modularity metric. To assess the accuracy of our algorithm, we first carry out experiments on two popular real-world networks, Zachary's karate club network, and Dolphin, with more details.

Initially, we have considered Karate ($E1$) for analyzing. The real-world Karate network originally has two predefined communities, with colors blue and red, which is demonstrated in Figure 6. After testing our algorithm on this dataset, the result is illustrated in Figure 7. It can be seen that our proposed algorithm can detect two major communities like original communities. However, this algorithm finds a small community for just two nodes 25 and 26. Since this minor community has just two nodes if it is considered inside the community with node 34, it is like an original network. Nevertheless, LPA-CNP, BGLL, LPA, and CNM can detect three communities in the Karate network. The modularity measure is obtained for this algorithm in Table 5. As it is seen in this table, the modularity value for our algorithm is $Q = 0.392$ which is better than a CNM algorithm with $Q = 0.380$, BGLL with $Q = 0.381$, LPA-CNP with 0.302, and LPA with $Q = 0.391$. However, it is a little lower than LPAm+ with $Q = 0.410$. In general, our algorithm can detect communities with good modularity and higher stability.

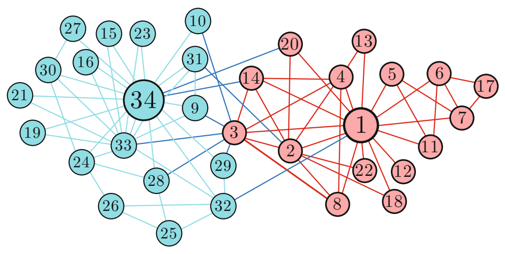

Figure 6 The real community structures of karate club

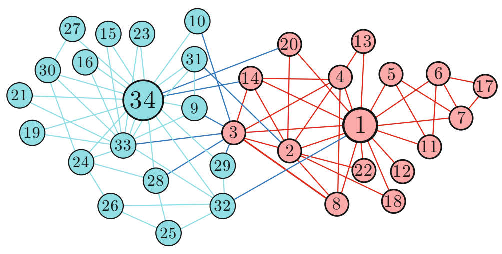

Figure 7 The result of our algorithm on karate club

The second real network is Dolphin network ($E2$) which is represented in Figure 8. The original Dolphin network has two communities. After running our algorithm on Dolphin dataset, three communities were detected (Figure 9) which is better than BGLL (5 communities), CNM (4 communities), LPA-CNP (3 communities), LPAm+ (3 communities) and LPA (4 communities). Furthermore, the modularity metric (Q) is calculated in Table 5 for our proposed method and other compared algorithms. The evaluation of modularity metric in this dataset reveals that our proposed method (with $Q = 0.552$) outperform CNM, BGLL, LPA-CNP, LPAm+ and LPA algorithms. All other algorithms are weaker than our method in both discovered number of communities and modularity value.

In addition, in Football dataset ($E3$), our proposed algorithm can detect 13 communities for 12 football conferences. In comparison with other algorithms, the obtained results are satisfactory and better. The analysis of modularity metric also shows that our proposed method has a better modularity value than LPA, CNM and BGLL methods. In addition, it is comparable to the LPAm+ algorithm.

In Political Books dataset ($E4$), the obtained results for our proposed algorithm is nearly similar to other compared methods. It is better than CNM, LPA-CNP and LPA methods. Furthermore, its modularity value is nearly similar to LPAm+ and BGLL algorithms. In datasets $E5$ to $E10$, we discuss the modularity metrics only since the number of communities is unknown. The evaluation of modularity value proves that our algorithm outperforms other compared algorithms on datasets $E6$, $E7$, $E9$ and $E10$ due to its Q value. All other algorithms have a remarkably weaker performance on these datasets. Moreover, in datasets $E6$ and $E8$, the modularity value for our algorithm is better than LPA-CNP and LPA. However, it is a little lower than BGLL and CNM.

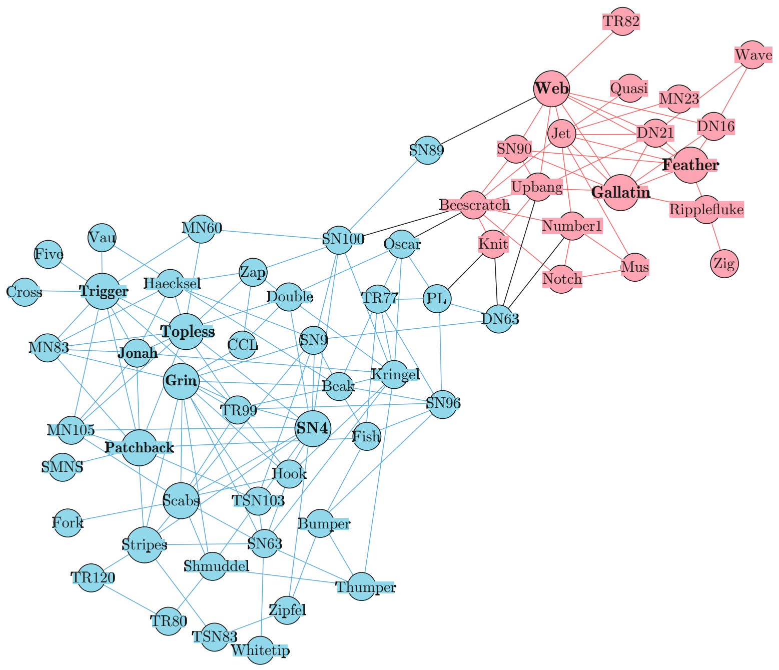

Figure 8 The real community structures of Dolphins

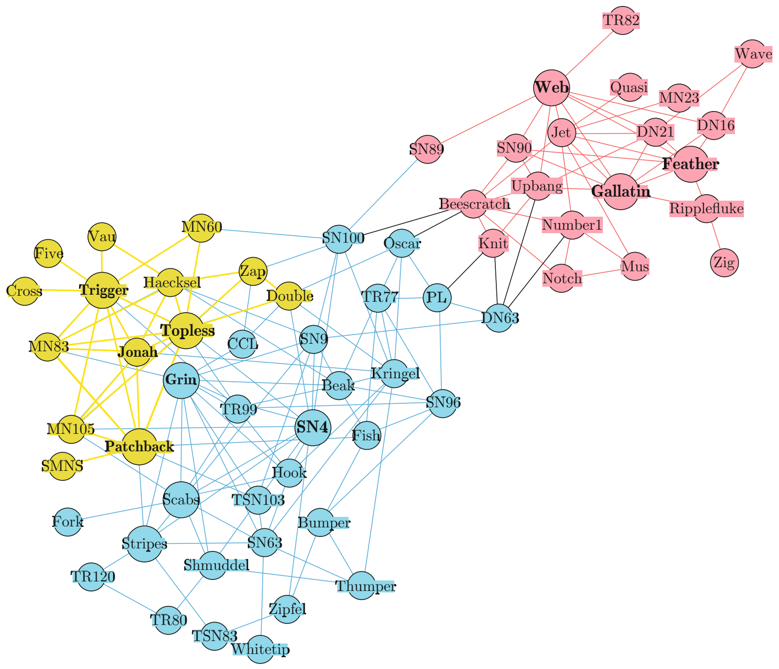

Figure 9 The result of our algorithm on Dolphins

Consequently, the experimental results of this paper show that, generally speaking, CNM has poor performance in finding communities and modularity metrics. In addition, LPA has a significant fluctuation problem in the obtained results that make it an unstable method. Also, LPA, in general, has a weaker performance than our algorithm, BGLL, and LAPm+. Moreover, our proposed method outperforms BGLL and LPAm+ algorithms in 8 out of 10 real datasets. The experimental results mostly show that our method has better efficiency with higher modularity than other algorithms.

Table 5 Result of modularity obtain on algorithms in real data set

Data	CNM		BGLL		LPA-CNP		LPAm+		LPA		Proposed	
Sets	Q	Num.	Q	Num.	Q	Num.	Q	Num.	Q	Num.	Q	Num.
$E1$	0.380	3	0.381	3	0.302	3	0.410	2	0.391 ±0.25	3±1	0.392	3
$E2$	0.495	4	0.418	5	0.466	3	0.524	3	0.410 ±0.280	4±1	0.552	3
$E3$	0.549	6	0.603	10	0.606	12	0.603	12	0.571 ±0.180	10±3	0.602	13
$E4$	0.501	4	0.520	4	0.451	8	0.526	6	0.481 ±0.141	5±1	0.520	6
$E5$	0.438	5	0.441	5	0.431	4	0.445	4	0.291 ±0.230	4±1	0.402	4
$E6$	0.408	14	0.440	10	0.429	12	0.452	14	0.215 ±0.159	5±1	0.473	14
$E7$	0.489	17	0.541	13	0.480	25	0.582	33	0.500 ±0.200	12±5	0.542	39
$E8$	0.955	403	0.959	406	0.932	441	0.941	319	0.901 ±0.210	454±6	0.953	334
$E9$	0.934	40	0.936	40	0.810	694	0.928	536	0.800 ±0.100	488±5	0.941	769
$E10$	0.601	49	0.6608	46	0.603	209	0.638	196	0.4108±0.29	380±15	0.691	215

NMI is another comparison metric for evaluating the accuracy rate found in the known community (Equation (2)). NMI produces the values between 0 and 1, with 1 corresponding to a faultless matching. The obtained results for NMI measure are shown in Table 6 for all algorithms with optimal parameters.

Table 6 Result of modularity obtain on algorithms in real data set

Data	Proposed			LPA			BGLL			Infomap			CNM		
Sets	NMI	N_d	N_c	NMI	N_d	N_c	NMI	N_d	N_c	NMI	N_d	N_c	NMI	N_d	N_c
($E1$)	0.68	3	2	0.581±0.110	3±1	2	0.59	3	2	0.69	3	2	0.69	3	2
($E2$)	0.56	3	2	0.491±0.091	4±1	2	0.48	5	2	0.53	6	2	0.55	4	2
($E3$)	0.82	13	12	0.654±0.124	10±3	12	0.88	10	12	0.97	12	12	0.75	6	12
($E4$)	0.53	6	3	0.494±0.081	5±1	3	0.51	4	3	0.49	6	3	0.53	4	3

The first evaluation is between our algorithm and original LPA. The results show that our algorithm outperforms LPA in all datasets. The NMI of our proposed method is also higher than BGLL in all datasets, except for the $E3$ dataset. Furthermore, our algorithm has higher

NMI than Infomap in datasets $E2$ and $E4$. It also is better than Fastgreedy in datasets $E2$, $E3$, and $E4$. In addition, the NMI of our algorithm is nearly similar to Infomap and Fastgreedy in data set $E1$. Comparing the NMI measure and number of detected communities reveals that our algorithm outperforms other algorithm.

6 Conclusion

In the recent decade, community detection algorithms based on label propagation strategy have become a hot research topic in the realm of complex networks. However, these methods have some major disadvantages in performance and stability metrics. In this paper, we proposed an improved version of LPA algorithm by changing the initial node selection strategy, the updating order and rule update strategy. The algorithm firstly calculates the link and node strength values and ranks the node in the descending order based on link strength value. It initially selects the nodes which have the highest node strength value. Then, it processes label updating based on the highest link strength value among neighbors. In tiebreak state, it uses the highest node strength among neighbors. This modification helps the traditional LPA algorithm avoid random selection and label propagation. Experiments were tested on synthetic benchmark networks and several real-world networks. The obtained results show that our algorithm can detect communities in networks and achieve higher accuracy and better stability. In the future, we believe that it is helpful to apply this method when analyzing multidimensional real networks like Flicker, Delicious, and IMDB networks. Furthermore, this method can clearly discriminate hub and core from the bridge and periphery nodes. It, therefore, can be used for influence maximization problem in complex networks.

References

[1] Newman Mark E J, The structure and function of complex networks, *SIAM Review*, 2003, **45**(2): 167–256.

[2] Scott J, *Social Network Analysis*, 4th Edition, SAGE Publications, London, UK, 2017.

[3] Dorogovtsev S and Mendes J, *Evolution of Networks: From Biological Networks to the Internet and WWW*, Oxford University Press, Oxford, 2003.

[4] Andrei B, Ravi K, Farzin M, et al., Graph structure in the web, *Computer Networks*, 2000, **33**(1): 309–320.

[5] Roger G and Nunes A L A, Modeling the world-wide airport network, *The European Physical Journal B-Condensed Matter and Complex Systems*, 2004, **38**(2): 381–385.

[6] Roger G and Nunes A L A, Functional cartography of complex metabolic networks, *Nature*, 2005, **433**(7028): 895–900.

[7] Romualdo P S and Alessandro V, *Evolution and Structure of the Internet: A Statistical Physics Approach*, Cambridge University Press, Cambridge, 2007.

[8] Bullmore E and Sporns O, Complex brain networks: Graph theoretical analysis of structural andfunctional systems, *Nature Reviews Neuroscience*, 2009, **10**(4): 312–312.

[9] Jackson M O, *Social and Economic Networks*, Princeton University Press, Princeton, 2010.

[10] Réka A and Albert-László B, Statistical mechanics of complex networks, *Reviews of Modern Physics*, 2002, **74**(1): 47.

[11] Costa L da F, Francisco A R, Gonzalo T, et al., Characterization of complex networks: A survey of measurements, *Advances in Physics*, 2007, **56**(1): 167–242.

[12] Santo F, Community detection in graphs, *Physics Reports*, 2010, **486**(3): 75–174.

[13] Michele C, Fosca G, and Dino P, A classification for community discovery methods in complex networks, *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 2011, **4**(5): 512–546.

[14] Shi G, Fabio P, Matthew C, et al., Controllability of structural brain networks, *Nature Communications*, 2015, **6**: 8414, DOI: 10.1038/ncomms9414.

[15] Michelle G and Newman Mark E J, Community structure in social and biological networks, *Proceedings of the National Academy of Sciences*, 2002, **99**(12): 7821–7826.

[16] Nandini R U, Réka A, and Soundar K, Near linear time algorithm to detect community structures in large-scale networks, *Physical Review E*, 2007, **76**(3): 036106.

[17] Leung Ian X Y, Pan H, Pietro L, et al., Towards real-time community detection in large networks, *Physical Review E*, 2009, **79**(6): 066107.

[18] Liu X and Tsuyoshi M, Advanced modularity-specialized label propagation algorithm for detecting communities in networks, *Physica A: Statistical Mechanics and Its Applications*, 2010, **389**(7): 1493–1500.

[19] Šubelj L and Bajec M, Unfolding communities in large complex networks: Combining defensive and offensive label propagation for core extraction, *Physical Review E*, 2011, **83**(3): 036103.

[20] Filippo R, Claudio C, and Federico C, et al., Defining and identifying communities in networks, *Proceedings of the National Academy of Sciences of the United States of America*, 2004, **101**(9): 2658–2663.

[21] Santo F, Vito L, and Massimo M, Method to find community structures based on information centrality, *Physical Review E*, 2004, **70**(5): 056104.

[22] Newman M E, Fast algorithm for detecting community structure in networks, *Physical Review E*, 2004, **69**(6): 066133.

[23] Clauset A, Newman Mark E J, and Moore C, Finding community structure in very large networks, *Physical Review E*, 2004, **70**(6): 066111.

[24] Blondel V D, Guillaume J L, Lambiotte R, et al., Fast unfolding of communities in large networks, *Journal of Statistical Mechanics: Theory and Experiment*, 2008, **2008**(10): P10008.

[25] Fortunato S and Barthélemy M, Resolution limit in community detection, *Proceedings of the National Academy of Sciences*, 2007, **104**(1): 36–41.

[26] Pons P and Latapy M, Computing communities in large networks using random walks, *ISCIS*, 2005, **3733**: 284–293.

[27] Shon H S, Kin S, Rhee C S, et al., Clustering DNA Microarray Data by MCL Algorithm, ISMB, 2007.

[28] Rosvall M and Bergstrom C T, Maps of random walks on complex networks reveal community structure, *Proceedings of the National Academy of Sciences*, 2008, **105**(4): 1118–1123.

[29] Sun H L, Liu J, Huang J B, et al., CenLP: A centrality-based label propagation algorithm for

community detection in networks, *Physica A: Statistical Mechanics and Its Applications*, 2015, **436**: 767–780.

[30] Barber M J and Clark J W, Detecting network communities by propagating labels under constraints, *Physical Review E*, 2009, **80**(2): 026129.

[31] Xie J R and Szymanski B K, Labelrank: A stabilized label propagation algorithm for community detection in networks, *Network Science Workshop* (*NSW*), 2013 *IEEE* 2*nd*, 2013, 138–143.

[32] Lou H, Li S H, and Zhao Y X, Detecting community structure using label propagation with weighted coherent neighborhood propinquity, *Physica A: Statistical Mechanics and Its Applications*, 2013, **392**(14): 3095–3105.

[33] Liang Z W, Li J P, Yang F, et al., Detecting community structure using label propagation with consensus weight in complex network, *Chinese Physics B*, 2014, **23**(9): 098902.

[34] Lin Z, Zheng X L, Xin N, et al., CK-LPA: Efficient community detection algorithm based on label propagation with community kernel, *Physica A: Statistical Mechanics and Its Applications*, 2014, **416**: 386–399.

[35] Liu X, Xie Z, and Yi D Y, Community detection by neighborhood similarity, *Chinese Physics Letters*, 2012, **29**(4): 048902.

[36] Scripps J, Tan P N, and Esfahanian A H, Node roles and community structure in networks, *Proceedings of the* 9*th WebKDD and* 1*st SNA-KDD* 2007 *Workshop on Web Mining and Social Network Analysis*, 2007, 26–35.

[37] Žalik K R, Maximal neighbor similarity reveals real communities in networks, *Scientific Reports*, 2015, **5**: 18374.

[38] Lancichinetti A, Fortunato S, and Radicchi F, Benchmark graphs for testing community detection algorithms, *Physical Review E*, 2008, **78**(4): 046110.

[39] Zachary W W, An information flow model for conflict and fission in small groups, *Journal of Anthropological Research*, 1977, **33**(4): 452–473.

[40] Lusseau D, Schneider K, Boisseau O J, et al., The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations, *Behavioral Ecology and Sociobiology*, 2003, **54**(4): 396–405.

[41] Newman M E and Girvan M, Finding and evaluating community structure in networks, *Physical Review E*, 2004, **69**(2): 026113.

[42] Gleiser P M and Danon L, Community structure in jazz, *Advances in Complex Systems*, 2003, **6**(4): 565–573.

[43] Jordi D and Alex A, Community detection in complex networks using extremal optimization, *Physical Review E*, 2005, **72**(2): 027104.

[44] Roger G, Leon D N, and Albert D G, Self-similar community structure in a network of human interactions, *Physical Review E*, 2003, **68**(6): 065103.

[45] Newman M E J, Finding community structure in networks using the eigenvectors of matrices, *Physical Review E*, 2006, **74**(3): 036104,

[46] Watts D J and Strogatz S H, Collective dynamics of "small-world" networks, *Nature*, 1998, **393**(6684): 440–442.

[47] Leon D N, Albert D G, Jordi D, et al., Comparing community structure identification, *Journal of Statistical Mechanics: Theory and Experiment*, 2005, **2055**(9): P09008.

[48] Network data, www-personal.umich.edu/ mejn/netdata/.