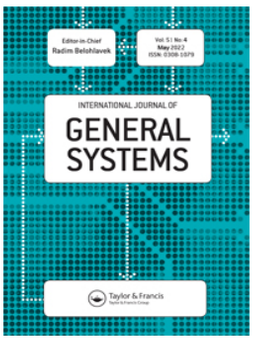

International Journal of General Systems

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/ggen20

A fast community detection algorithm using a local and multi-level label diffusion method in social networks

Asgarali Bouyer, Khatereh Azad & Alireza Rouhi

To cite this article: Asgarali Bouyer, Khatereh Azad & Alireza Rouhi (2022) A fast community detection algorithm using a local and multi-level label diffusion method in social networks, International Journal of General Systems, 51:4, 352-385, DOI: 10.1080/03081079.2022.2025794

To link to this article: https://doi.org/10.1080/03081079.2022.2025794

Published online: 31 Jan 2022.

INTERNATIONAL JOURNAL OF GENERAL SYSTEMS
2022, VOL. 51, NO. 4, 352–385
https://doi.org/10.1080/03081079.2022.2025794

A fast community detection algorithm using a local and multi-level label diffusion method in social networks

Asgarali Bouyer, Khatereh Azad and Alireza Rouhi

Department of Software Engineering, Azarbaijan Shahid Madani University, Tabriz, Iran

ABSTRACT

One of the popular categories of community detection methods are label propagation-based algorithms. Label propagation-based algorithms use local criteria and have a near-linear time complexity. However, these algorithms have problems such as low accuracy, instability, and high computational time in comparison with other local methods. This article presents a fast and simple label diffusion method (FSLD), using local criteria to discover communities accurately in large-scale networks. In FSLD method, community formation is initially started from a low-degree periphery node and then it diffuses its label from outer to inner side of community in a multi-level way. In next step, using a label updating step, all nodes from high-degree to low-degree have the potential to update and finalize their label to obtain initial communities. The experimental results reveal the higher accuracy and performance of the proposed FSLD algorithm in comparison to other state-of-the-art algorithms.

ARTICLE HISTORY
Received 2 September 2021
Accepted 30 November 2021

KEYWORDS
Social networks; community detection; local similarity; multi-level label diffusion

1. Introduction

By exploring in the real-world, many complex systems such as social networks, Internet, the World Wide Web (WWW), protein–protein interactions, and several Metabolic networks are found. Such systems can be modeled as complex networks in which entities are represented by nodes and their relationships are called edges or links (Berahmand and Bouyer 2019). For example, the WWW is a network of web pages interconnected by links. Social networks are represented by individuals as their nodes and the underlying interrelationships are represented by edges. In addition, biological networks are represented by biochemical molecules as nodes and the reaction between them by edges.

Today's modern networks appear to be large-scale, and technology advancements such as the Internet and WWW have led to the rapid growth of these networks. Among these networks, social networks rapidly have been grown in recent decade. One of the main features in complex networks is to have community structures. A community, also called cluster or module, are defined as a groups of nodes which have high edge density intra-groups and low edge density inter-groups (Danon et al. 2005; Boccaletti et al. 2006).

CONTACT Asgarali Bouyer a.bouyer@azaruniv.ac.ir

Community detection in social networks is one of the branches of complex network analysis. It helps us to analyze the structure of existing communities and the role of important nodes inside communities (Bouyer and Roghani 2020). Community detection can be used in a variety of contexts (Ren and Wang 2014). For example, clustering web clients who have similar interests and are geographically close to each other may improve the quality of services provided to them because each cluster of clients can be served by a server computer. In the case of marketing, by finding communities, it is possible to identify the interest of each community on specific product items and suggest the items needed by that community according to their favorites. The communities' structure can be used in reality epidemic spreading (Ren and Wang 2014). In social networks, by identifying the friend groups, the people interrelationships can be analyzed. In protein–protein networks, proteins with similar functions can be identified.

In recent years, various algorithms were proposed to discover communities in complex networks. These algorithms are divided into global and local categories in terms of community detection. Global methods require general information across the network. Thus, many community detection algorithms are successful in small networks with hundreds or thousands of nodes but have challenges in large-scale networks with millions or billions of nodes and edges (Taheri and Bouyer 2020). Many of the global methods cannot be implemented in the large networks due to their high time complexity. For example, the community detection method based on the edge's flow information and shortest path computing with n vertices and m edges needs the $O(m^2n)$ time complexity (Girvan and Newman 2002) that is impractical on medium and large-scale networks. In addition, processing all of the network's data may require large storage and frequent access, which is expensive in terms of storage (Tasgin and Bingol 2019). A local community detection approaches have required performance in large networks due to using local information around some core nodes as well as the local similarities such as the Jaccard index (Niwattanakul et al. 2013), the Salton index (Salton 1975), the Sorensen Index (Aghaalizadeh et al. 2021), and so on. Most of local methods have near-linear time complexity such as $O(m)$ or $O(nk)$ which enable them to use in large-scale networks despite their lower accuracy than global algorithm (Pan et al. 2010). Therefore, one of the major challenges in local algorithms is to reach a high accuracy in identifying communities with preserving the low time complexity (Fortunato 2010). Another challenge is to consider the importance of nodes to select and expanding communities core (Fortunato, Latora, and Marchiori 2004). One of the most popular local methods with almost linear time complexity is the Label Propagation Algorithm (LPA) (Raghavan, Albert, and Kumara 2007). However, due to the random behavior of the node selection and the weakness of the label update strategy in LPA, it does not provide the stable and adequate results. However, there are many attempts to offer an improved version of LPA with better performance. There are other fast community detection algorithms such as Louvain algorithm that is not used in large networks due to their inappropriate accuracy (Blondel et al. 2008).

This article presents a new method using the method of label diffusion from marginal nodes by considering local criteria and similarities to discover communities with low time complexity. In addition, considering the importance of nodes and the influence of each node neighbors can increase the accuracy of finding communities. Therefore, in the proposed algorithm, first the nodes are grouped based on their degrees. Then it is started with the group of degree 2 nodes and after assigning the label to all nodes of degree 2,

the nodes of degree 3, 4, and so on are evaluated to assign proper label. This operation continues until all nodes in the graph are labeled. In the next step, initial communities are formed. Finally, some of the initial communities is merged into other communities to obtain the final communities. The proposed algorithm is evaluated on the real and artificial networks and experiments show that this algorithm has a low time complexity and can detect communities with better quality and stability.

The remainder of this article is organized as follows. In Section 2, some recent studies on community detection are discussed. In Section 3, the proposed algorithm is presented. In Section 4, experiments and evaluation of the results on the real and artificial networks are reviewed. Finally, Section 5 concludes the article.

2. Related work

In recent years, many algorithms for community detection have been proposed. These algorithms can be classified into two categories: globally community detection algorithms and locally community detection algorithms. Global methods have higher time complexity in general and their performance is very poor in the case of large-scale networks. Alternatively, community detection algorithms can operate locally (or semi-locally). Local algorithms take into account the impact of the neighborhood only at the first level, while semi-local algorithms use the neighborhood effect at the second and the third level of the neighborhood. The global methods include graph partitioning-based methods, hierarchical clustering-based methods, traditional clustering methods, and the modularity optimization-based methods, to name a few (Girvan and Newman 2002; Kernighan and Lin 1970; Newman 2004). Local community detection algorithms include label propagation-based, diffusion-based, core-expanded-based, and other local similarity-based methods.

On the other hand, we can classify algorithms into two main categories: overlapping and non-overlapping community detection algorithms. Overlapping communities are possible if a node is a member of more than one community.

Traditional clustering methods such as C-means and K-means have also been used in this field. One of the proposed approaches is to identify overlapping communities based on the C-means clustering method (Lei, Zhou, and Shi 2019). In this method, two sets of fuzzy and rough are used. Another overlapping community detection method is in bipartite networks by extracting some key bi-communities and free nodes (Cui and Wang 2014). In an article (Cui and Wang 2016), an algorithm is proposed to detect one-mode community structures in bipartite networks, which one-mode community structures are weighted. This algorithm mainly includes two phases: the first phase of this algorithm is that the original bipartite network is projected into two weighted one-mode networks basing on the bipartite clustering triangular. The second phase of this algorithm is respectively detecting one-mode community structures from two weighted unipartite networks. In addition, a novel method using a new and asymmetric parameter is used to measure the intimate relationship among and within nodes or communities, which effectively can define relationships (Wang and Qin 2016). In the final step, they have merged the initial communities to reveal final communities and also get the overlapping nodes in bipartite networks. The LICOD method uses the leader-driven concept to create communities around leader nodes (Yakoubi and Kanawati 2014). To select the leader nodes, nodes with higher degree than

their neighbors are selected. Then, leaders are considered members of a community with a certain percentage of common neighbors. Eventually, each node joins the relevant community. Moradi et al. use similarity criteria between nodes to improve seed node selection in seed selection approaches (Moradi, Olovsson, and Tsigas 2014). The SLPA method (Xie and Szymanski 2012) is a label propagation-based method for overlapping community detection that extends labels according to dynamic interaction rules and maintains the label distribution in the memory of each node. In Li et al. (2017), a method called Stepping LPA-S is proposed where labels are published based on similarity. In addition, this algorithm divides networks using a stepping framework and uses the proposed evaluation function to select the final unique partition. Wang and Li (2013) proposed an overlapping algorithm using the core-vertex and the intimate degree between the community and its neighboring vertices based on this fact that a node with a noticeably large degree is likely the core of a community, and selected these nodes as the core-vertices of communities. Cui et al. proposed an overlapping community detection algorithm based on the maximal sub-graphs and the clustering coefficient of two neighboring communities. In their proposed ACC algorithm, all the maximal sub-graphs first extracted and then merged with neighboring maximal sub-graphs to obtain the communities using clustering coefficient of two neighboring sub-graphs (Cui, Wang, and Li 2014). Eustace et al. utilized the overlapping neighborhood ratio to assign nodes to their communities (Eustace, Wang, and Cui 2015b). Then, they utilized matrix factorization to assign nodes into their corresponding community structures. BASH (Cui, Wang, and Eustace 2014) is another overlapping community detection algorithm that uses the maximal sub-graphs and the belonging degrees. In addition, there are some other methods for community detection which use the policy of calculating local similarity and joining into the appropriate community (Eustace, Wang, and Cui 2015a; Wang, Yin, and Wang 2018; Zarandi and Rafsanjani 2018; Pan et al. 2019).

Non-overlapping network is familiar with nodes of community in the network in which they do not have relation with another network. The most popular and traditional non-overlapping graph partitioning algorithm is spectral bisection and Kernighan–Lin (Kernighan and Lin 1970; Barnes 1982). The Kernighan–Lin algorithm is one of the first proposed methods which is still used in combination with other techniques. Owing to the high time complexity of the algorithm and the low quality of obtained partitions, the algorithm is only used in small networks. Yin et al. (2015) proposed an algorithm for community detection based on hierarchical clustering called CDHC. This algorithm first creates initial communities from the global central nodes, then it expands the initial communities according to the strength of the edge between nodes and communities and finally merges weak and small communities into large communities. Furthermore, a new approach based on the divide and agglomerate method was proposed in Liu and Ma (2019). Here, first each node selects its most similar neighbor and connects to that, and the initial partitions are obtained. Then, according to the criteria of the community and the principle of group integration, the final communities are determined. In addition, GN, CNM and Newman's greedy algorithms are traditional modularity optimization-based methods for non-overlapping community detection (Newman 2004; Newman and Girvan 2004; Clauset, Newman, and Moore 2004). In addition, Louvain algorithm is a modularity optimization-based method with fast detection (Blondel et al. 2008). In this method, by considering each node as a community, each node is placed in a neighboring community with the maximum increase in modularity and as a result the initial communities are

obtained. The resulting communities are merged to create communities with high modularity. The LCCD (Wang et al. 2017) is a local core-expanded method where core nodes are identified using the core density and the distances between nodes, and the subgraph density criteria are used to extend the core nodes. In addition, Berahmand, Bouyer, and Vasighi (2018) presented a local core-expanded non-overlapping approach to identify communities by weighting nodes. In the first step, by calculating the extended Jaccard similarity for each node, the node with the highest value is selected as the central node. Then, in the second step to expand the central node, a neighbor node is conditionally added to the initial communities. Finally, the two communities are merged with the largest edge weight. In other presented method, a local edge centrality for edge weighting was proposed (Li et al. 2019). In this method, by calculating the centrality of the local edge of each edge, larger edges are removed from the threshold. To obtain the final partitions, the two communities are merged if the maximum modularity is reached. In addition, authors in Gamgne Domgue, Tsopze, and Ndoundam (2020) have proposed a simple kernel scheme to improve the detection of communities in directed networks, through triad density and kernel degree. It focuses on kernels which are seed nodes centralizing information through their in-degree valuation.

On the other hand, non-overlapping label propagation-based methods were first proposed by Raghavan, Albert, and Kumara (2007) and quickly became popular due to their near-linear time complexity for non-overlapping community detection. However, it had several disadvantages. In Barber and Clark (2009), an extended LPA, called LPAm, is proposed using a change in the update rule to maximize the modularity criterion. LPAm cannot find the global optimum for community detection due to its stuck at the local maximum. To escape the local maximum, Liu et al. used a greedy multi-step algorithm to merge several pairs of communities at once. The combination of LPAm and this algorithm introduces another method based on label propagation and modularity called LPAm+ (Liu and Murata 2010). In LP-LPA method (Berahmand and Bouyer 2018), the node with the higher label effect is selected as the primary node. Then, to propagate the label, a label is selected from the neighbor nodes that have high edge strength. The G-CN method detects communities by identifying boundary nodes (Tasgin and Bingol 2019). Hosseini and Rezvanian (2020) propose an optimized version of LPA, called AntLP, using the similarity indexes and ant colony optimization. Bouyer and Roghani (2020) present a fast and accurate community detection algorithm, called LSMD, based on local information for discovering non-overlapped communities using a diffusion strategy. In the first step of the LSMD algorithm, the first label is assigned to a degree 1 node and its direct neighbor and neighbors of the second level. Then, the label is assigned to nodes with degree 2, 3, and so on, respectively. In the second step, the initial communities are merged to form the final communities using a simple and quick strategy. PLDLS is another diffusion-based method that uses a multi-factor node scoring measure with low time complexity for ranking and detecting core nodes (Roghani, Bouyer, and Nourani 2021). Then, it uses a new parallel and distributed method for diffusing labels of core nodes and a new label selecting method. Finally, it provides a new and fast candidate-based merge method to obtain dense communities. An improved label propagation algorithm named LPA-MNI is proposed in this study by combining the modularity function and node importance with the original LPA (Li et al. 2021b). LPA-MNI first identified the initial communities according to the value of modularity. Subsequently, the label propagation is used to cluster the remaining

nodes that have not been assigned to initial communities. Meanwhile, node importance is used to improve the node order of label updating and the mechanism of label selecting when multiple labels are contained by the maximum number of nodes. A new community detection algorithm DS-LPA based on density peak clustering and label propagation is proposed in this study, which comprehensively considers the influence of nodes and their neighbors on local density, and calculates the contribution of each node by defining the neighborhood density, thereby improving the discrimination between the central node and other nodes (Li et al. 2021a). When calculating the minimum distance, Jaccard index is used to measure the similarity between nodes, and then the distance between nodes is calculated by similarity. Then a threshold is introduced to select the center point when calculating the minimum distance. After constructing the initial community based on the community center, the order of label is updated by calculating the information transmission power of the nodes, which supplements the label propagation rules of LPA and greatly reduces the randomness of the label propagation algorithm. DPNLP (Zarezadeh, Nourani, and Bouyer 2021) is new LPA-based method to fast detection of community structures in large-scale network. DPNLP has low execution time with adequate quality in detecting community structures. it uses a novel local distance-based measure and peripheral nodes label propagation algorithm for community detection.

3. The proposed method

The proposed fast and simple label diffusion method (FSLD) algorithm is categorized in label diffusion group such as LSMD and PLDLS methods. All methods of label diffusion diffuse the labels from main nodes to other nodes by finding the main proper nodes. In these methods, the nodes must be sorted based on an importance measure. However, the proposed method does not use any sorting algorithm. Here, the nodes are placed in different lists based on their degrees. For example, the first list includes degree 1 nodes; the second list contains degree 2 nodes; and so on. In the proposed method, nodes with degree 1 will not be considered until the last step. In the last step, they are labeled by the label on their connected node. As a result, the proposed method is faster than other label propagation methods. This method starts the label diffusion process with the list of degree 2 nodes and after assigning a label for all nodes of this list, it will continue labeling of other lists, respectively. The selected node in each step is named "*target node*". The target node diffuses its label to neighbor nodes. Initially, all nodes are labeled to zero (i.e. no label) and it means that the desired node does not belong to any community. This algorithm consists of four steps: *label diffusion*, *label updating*, *label diffusion to degree 1 nodes*, and *merging the initial communities*. General steps of the FSLD algorithm are shown in Figure 1. These steps are described in more detail in the following context.

3.1. Label diffusion step

First, the network nodes are classified based on their degrees into different lists. Initially, all nodes do not have a label. Then, this algorithm starts with degree 2 list to specify the community label for all nodes. This method also defines a counter, named "*Counter*", with initial value 0 that is considered as community label in next steps. The *Counter* value increases by assigning a new label. In other words, each *Counter* value shows the

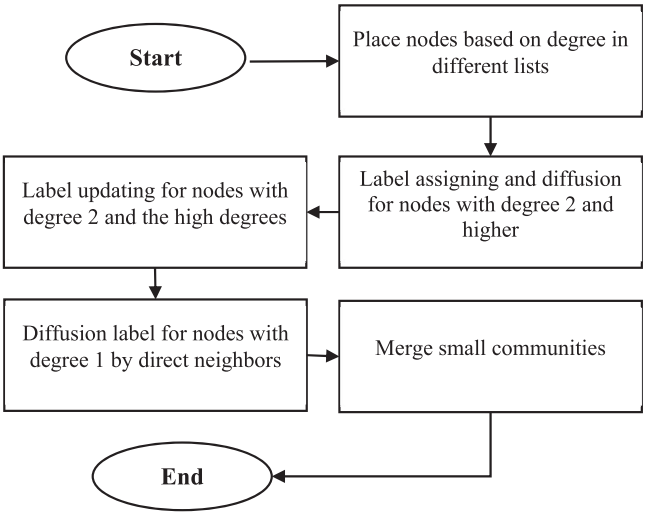

Figure 1. A flowchart of general steps for the proposed FSLD algorithm.

community label for node. First, an unlabeled node (zero value) is selected as the target node v from the list of degree 2 nodes. The label of the direct neighbors of the target node is considered. If all neighbors do not have a label (zero value), we calculate a node influence power using the degree and common neighbor features of the target node v with each of its neighbors by Equation (1):

$$DCN_{v,i} = deg_v + 2|N_v \cap N_i| \tag{1}$$

where node i is the neighbor of node v and N_i is its neighbors and N_v is the neighbors of target node v. The node with the highest similarity ($DCN_{v,i}$) is selected. Several scenarios are possible:

- If node v had more similarity to one of its neighbors than others, the *Counter* value of that node is increased by one, and this value is also assigned to the target node as the community label.
- If the same similarity is observed between node v and some of its neighbor nodes, a higher degree node is selected from them and its *Counter* value is increased that shows the assigned label for that node and also this *Counter* value is assigned to the target node as the community label.

However, if some of the neighbors of the target node had a label (non-zero label), two different cases can occur:

- Each of the neighbors has a unique label. In this case, node v selects the node from its neighbors with the highest similarity according to Equation (1) and checks if its neighbor had a label (non-zero), the same label is assigned to the target node as the community label.

- At least two neighbors have equal and non-zero labels. Here, we check the number of repetitions of each label in the neighborhood of node v. If a non-zero label has the higher frequency, this label is assigned to the target node v. However, if the zero-label has the highest frequency in the neighborhood of node v, we calculate the similarity of node v with an unlabeled neighbor node with the highest degree and a labeled neighbor node with the highest degree, according to Equation (1). If node v had more similarity to the labeled node, it takes its label. Otherwise, the *Counter* of the neighbor node is increased by one and the value is assigned as the community label to the target node like its neighbor. In addition, if there are several labels with the same maximum frequency, the neighbors with the highest degree are selected separately for each of these different labels. Then, the similarity of the target node with each of these neighbors is calculated using Equation (1). After that, the neighbor with the highest similarity are selected and the label of that neighbor is assigned to the target node as the community label. However, if the label was zero, according to the mentioned procedure, a new label will be created by increasing the *Counter* and assigned to the target node and the desired neighbor.

After considering the list of degree 2 nodes, the list of degree 3, degree 4, and so on are examined, respectively. At last, all nodes with degree 2 and higher degrees will get their community labels. Algorithm 1 shows this initial label assigning process for different lists.

3.2. Label updating step

After diffusing the initial labels to nodes with degree 2 and higher, the label updating phase is performed. This step is more important due to nearly finalizing the label of nodes in initial communities. Unlike the previous step, this step starts the label updating with the nodes of high-degree list to low-degree list. At first, it selects a node from highest degree list as the target node. Then the node influence power of the target node is calculated using Equation (1) to select a neighbor with the most similarity to the target node. If there exists a neighbor node with the highest *DCN* value and its label was different from the target node label, we assign that neighbor's label to the target node as the community label, otherwise the target node label is not updated. However, if we had more than one neighbor with the highest *DCN* value (equal conditions), we select a neighbor with highest degree and if the neighbor's label was different from the target node label, we assign that neighbor's label to the target node as the community label, otherwise the label is not updated for the target node. In the same way, we update the labels of all nodes in different lists. Algorithm 2 shows the label updating step for nodes with degree 2 and the higher.

3.3. Label diffusion to degree 1 nodes

After diffusing and updating the label for degree 2 nodes and higher, the list of nodes with degree 1 is checked to assign label of their connected node. Nodes with degree 1 are only connected to one neighbor. Therefore, from the list of nodes with degree 1, a node is selected as the target node. The target node receives the community label from its labeled neighbor. In this way, all nodes with degree 1 simply receive a community label. Needless to mention that this step must be performed after label updating step because its stability

Algorithm 1. Label assigning and diffusion for nodes with degree 2 and higher in FSLD algorithm.

Input: Network nodes grouped by their degrees
Output: All labeled degree 2 nodes and higher
1. Group nodes based on their degrees
2. Initialize each node label to 0
3. Counter ← 0
4. Repeat step 4 for each unlabeled degree 2 node and higher
- Select a target node which has not diffused label previously
- **If** target node label $==0$ // Check its neighbor's label
- **If** label of all neighbors $==0$
  - Calculate the target node's influence power with its neighbor using Eq. (1)
  - Select a neighbor with the highest similarity
  - *Counter* ← *Counter* + *1*
  - Assign the value of *Counter* as the label of selected neighbor and the target node
- else **If** label of all neighbors! $=0$
  - **If** there is a single node from each label
    - Calculate the target node's similarity with each of its neighbor by $DCN_{v,i}$
    - Select the neighbor with the highest similarity (in equal condition, Select a neighbor with the highest degree)
    - Label (target node) ← Counter value of this neighbor
  - **else if** there are 2 or more nodes from each label
    - Select the label with the highest frequency
    - **If** number (label with the highest frequency $==1$)
      - Assign this label as the community label for the target node
    - **else If** number (label with the highest frequency >1)
      - Select a neighbor with the highest degree for each label
      - Calculate the target node's influence power with each selected neighbor by $DCN_{v,i}$
      - Select a neighbor with the highest similarity
      - Label (target node) ← Counter value of this neighbor
- **else If** label of some neighbors! $=0$
  - Select the label with the highest frequency (in equal condition, Select a neighbor with the highest degree)
  - **If** label $==0$
    - Select a neighbor $i1$ with the highest degree and label! $=0$
    - Select a neighbor $i2$ with the highest degree and label $==0$
    - **If**$DCN_{v,i1} > DCN_{v,i2}$
      - Assign this label of node $i1$ as the community label for the target node
  - **else** - *Counter* ← *Counter* + *1*
    - Assign the value of *Counter* as the label of selected neighbor and the target node
  - **else** If label! $=0$
    - Assign this label as the community label for the target node
5. End

and running time considerably is affected in this case. At the end of this step, the nodes that have the same label are placed in the same community and thus the initial communities are formed.

3.4. Merge step

During the previous steps, initial communities are discovered. However, some of the initial communities should be merged to form final communities. Unlike other methods that use the modularity criterion to merge initial communities, this algorithm uses a new and fast method to merge the obtained initial communities. First, we calculate the average size of communities using Equation (2):

$$Avg = \frac{N}{|C|} \tag{2}$$

Algorithm 2. Label updating step for nodes with degree 2 and the high degrees

Input: Labeled all 2 and higher degree nodes and lists
Output: All updated labels
1. Start from high-degree list to lower degree one's
2. Repeat step 2 for each node in lists
- Select a target node which its label has not updated
- Calculate the target node's similarity with each neighbor by $DCN_{v,i}$
- Select the neighbor with the highest DCN value
- **If** there is only one neighbor with the highest DCN value
- **If** label (target node) $=$ $=$ label (selected neighbor)
- Continue (no update)
- **else**
- Label (target node) $\leftarrow$ Label (selected neighbor)
- **else**// *equal conditions*
- Select a neighbor with the highest degree
- **If** label (target node) $=$ $=$ label (selected neighbor)
- Continue (no update)
- **else**
- Label (target node) $\leftarrow$ Label (selected neighbor with the highest degree)
3. End

where N represents the number of nodes and C represents the number of initial communities. Communities smaller than Avg are considered small communities and communities larger than Avg are considered large communities. For each small community, we select the node with the highest degree. The goal is to find large communities around each small community. For this purpose, we examine and select the large communities, where the selected node neighbors have been scattered. Thus, a node with the highest degree from each large community is selected as candidate node. After that, for each pair of neighbor communities, the number of common neighbors of the node with the highest degree node from a small community and the candidate node of large community is calculated using Equation (1). Then, the node with the highest similarity is selected. At next, the internal density of a small community is calculated using the number of edges within the community. In addition, the external density of this community is also calculated using the number of edges between a small community and a large community. Then, the two communities are merged, if the Equation (3) be true:

$$\frac{\text{Inner edge}}{2} - \text{Outer edge} \leq 1 \tag{3}$$

The quality of this merge method is evaluated based on merging by modularity measure. The results show that using the condition of Equation (3), merging of weak and small community is performed with minimum time complexity. This step is applied for all small communities. In this way, some small communities are merged with the large ones. Now, for newly formed communities, the mentioned merge step is repeated once again to merge weak communities as far as possible. As a result, if some of weak communities were not merged in the first step, they possibly can be merged in the second step. Therefore, the final communities are formed after finishing this step. Algorithm 3 displays the steps of merging phase.

Algorithm 3. Merge of initial small and weak communities in FSLD algorithm

Input: Initial communities
Output: Final communities
1. Calculate the average size of communities
2. Determine small communities and large communities
3. Repeat step 3 for each small community
   - Select a node with the highest degree from small community
   - Find large communities around small community
   - Select a node with the highest degree from each large community
   - Calculate similarity of the selected node from small community with the selected node from each large community using Equation (1)
   - Among the candidate node of large communities, select the node with the highest similarity
   - Select the desired large community of that node that is neighbor to small community
   - Calculate the number of inner-edge density from small community
   - Calculate the number of outer-edge density from large community
   - **If** $\frac{\text{Inner edge}}{2} - \text{Outer edge} \leq 1$
     Merge small community with the considered large one
4. Repeat just once again the step 1–3 for new formed communities
5. End

3.5. Example

To help better understanding of FSLD algorithm, a step-by-step example is described using Zachary's Karate club dataset. The Karate data set consists of 34 nodes, 78 edges, and 2 communities. Each step of the algorithm is shown in a separate figure.

As mentioned previously, this algorithm starts with nodes of degree 2 list. Table 1 illustrates the node groups and their frequencies based on their degrees. The algorithm selects a node with degree 2. Thus, node 9 is selected as the target node. Nodes 2 and 33 are neighbors of node 9 with zero labels, so we calculate the similarity of node 9 with the nodes 2 and 33. The similarity of nodes 2 and 33 to node 9 is the same, so node 33 is selected due to having the highest degree and the *Counter* value is increased by one and the value is assigned as the community label to the target node (node 9) and its neighbor (node 33), which is shown in Figure 2. Then, another node is selected to diffuse the label. The node 12 is selected as the new target node. The neighbors of this node are nodes 0 and 3 with zero labels. Since the similarity of these neighbors to node 12 is the same, node 0 with the highest degree is selected and a new label is assigned to node 12 and node 0, which is shown in Figure 3. Now, node 14 is selected as the new target node. Nodes 32 and 33 are the neighbors of node 14 with different labels. Thus, node 33 is selected with the highest degree and its label is assigned to node 14. The next node to be selected as the target node is node 15. Similar to node 14, this node receives the community label of node 33 as its important neighbor. The result is shown in Figure 4. Node 16 is selected as the new target node which its neighbors are nodes 5 and 6 with zero label. Since the similarity of node 6 to the target node is the same as the similarity of node 5 to the target node, node 6 is selected with the highest degree and a new label is assigned to node 16 and its neighbor (node 6), which is shown in Figure 5. As discussed previously, this process is performed for other degree 2 nodes to receive a suitable community label based on their important neighbors. After assigning label to all degree 2 nodes, the result is shown in Figure 6.

After finishing the label diffusion to all degree 2 nodes, the label assigning is performed for list of degree 3 nodes. Therefore, node 4 is selected as the target node. Nodes 0, 6,

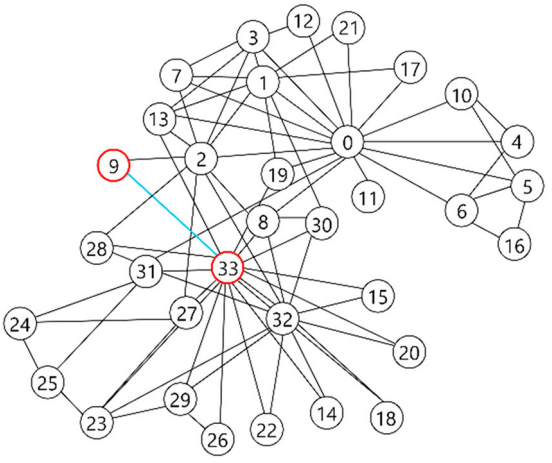

Figure 2. Diffuse the label to node 9 and node 3.

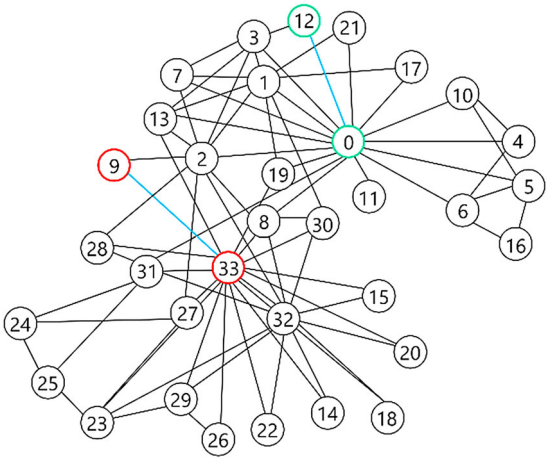

Figure 3. Diffuse the label to node 12 and node 0.

and 10 are the neighbors of node 4 that have different labels too. Therefore, we calculate the similarity of node 4 with each of the neighbors. Node 0 is selected with the highest similarity, and the label of node 0 is diffused to node 4. Then, node 10 is selected as the target node. Neighbors of node 10 are nodes 0, 4, and 5 where nodes 0 and 4 have the same labels. Thus, the same label is assigned to node 10. Node 19 is selected as the target node. Its neighbors are nodes 0, 1, and 33 which have different labels. We calculate the similarity of node 19 with each of its neighbors. The similarity of these neighbors to the node 19 is the same. The neighbor with the highest degree, node 0, is selected, and node 19 receives the community label from node 0. Next, the node 24 is selected as the target node. Neighbors of node 24 are nodes 25, 27, and 31 that have zero labels. By calculating the similarity of node 24 to each of its neighbors, node 31 is selected and a new label is assigned to the nodes 24 and 31. Then node 25 is selected as the target node. Two neighbors of node 25, nodes 24 and 31, have the same labels. Thus, this label is diffused to node 25. Finally, node

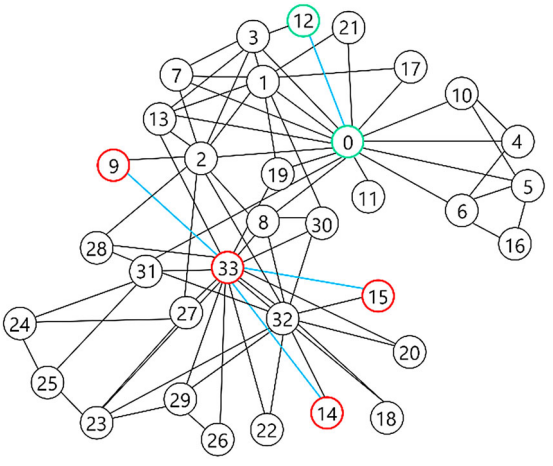

Figure 4. Diffuse the label to node 14 and node 15.

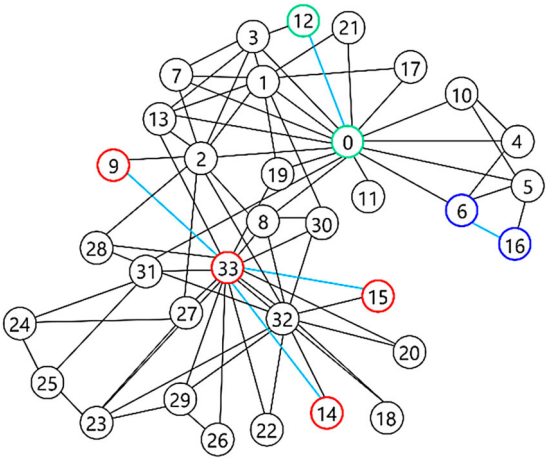

Figure 5. Diffuse the label to node 16 and node 6.

28 with degree 3 is selected as the target node. Node 28's neighbors are 2, 31, and 33 with different labels. We calculate the similarity of the target node with each of these neighbors. The similarity of node 31 to node 28 is the same as the similarity of node 33 to node 28 and has the highest value. Therefore, node 33 is selected with the highest degree and the label of this node is assigned to node 28. The obtained result is shown in Figure 7.

This process is carried out in the same way for the other list of nodes until the first step is completed, and all nodes are labeled. As a result, all remaining higher degree nodes will be assigned to a community label as well.

Now, the *label updating* step is started with higher degree to lower degree lists. The node with the highest degree is node 33, which is selected to update the label. Node 33 must choose the node with the highest similarity from its neighbors. Therefore, node 32 is selected as the neighbor of node 33 with the most similarity. The label of node 32 is the same as the label of node 33. Hence, the label of node 33 remains the same. Node 0

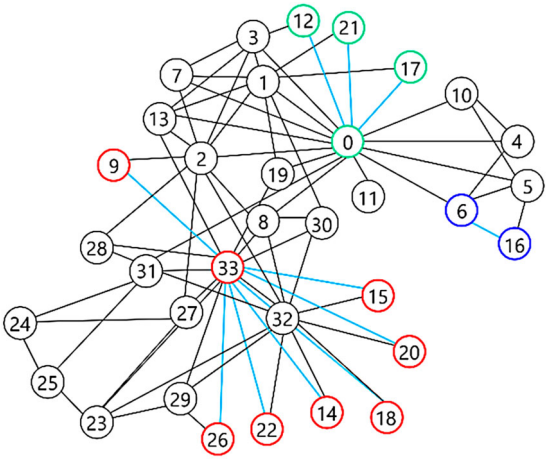

Figure 6. Assigned label for all nodes with degree 2.

Table 1. Zachary's Karate club node groups based on their degrees.

Group	Number of nodes
Degree 1	11
Degree 2	9, 12, 14, 15, 16, 17, 18, 20, 21, 22, 26
Degree 3	4, 10, 19, 24, 25, 28
Degree 4	5, 6, 7, 27, 29, 30
Degree 5	8, 13, 23
Degree 6	3, 31
Degree 9	1
Degree 10	2
Degree 12	32
Degree 16	0
Degree 17	33

is selected to update its label. The similarity of node 0 is calculated with its neighbors. Node 1 has the highest similarity to node 0. Here, the label of node 1 should be assigned to node 0. However, since the labels of the two nodes are the same, the label of node 0 does not change. Then node 32 with degree 12 is selected to update its label. By calculating the similarity of node 32 with its neighbors, node 33 is selected with the highest similarity. Since these two nodes have the same labels, the label of node 32 remains unchanged. Now node 2 is selected to update its label. Among its neighbors, node 0 has the most similarity to node 2. The label of node 0 is assigned to node 2 and the label is updated. Similarly, the label updating process is performed for other lists. The result of label updating step is shown in Figure 8.

After this step, it is required to diffuse the label for the nodes with degree 1. There is only node 11 with degree 1 in the Karate dataset. Node 11 receives the label of node 0 as its only neighbor. After this step, all nodes in the network are labeled and nodes having the same labels are placed in the same community. As shown in Figure 9, two communities are formed. In fact, the Karate network will have two final communities as well. Therefore, the Karate network does not need to apply the *merge* step of our proposed method.

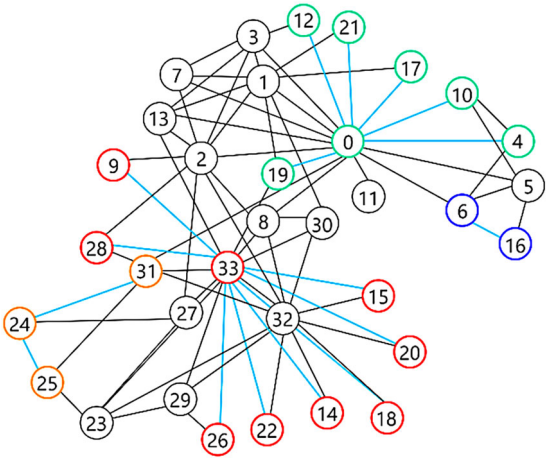

Figure 7. Label Diffusion to the nodes with degree 3.

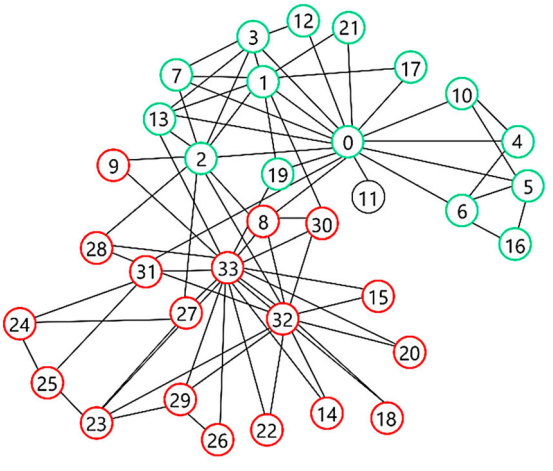

Figure 8. Update the label for all nodes with degree 2 and the high degrees.

3.6. Time complexity

The proposed FSLD algorithm consists of four separate steps represented by algorithms 1–4. The first step is diffusing label. In this step, only the degree 2 nodes and higher will be considered, which their number is denoted by n'. If all the nodes' number is equal to n, the number of nodes with degree 1 that is not considered in this step will be $n - n'$, and $n' < n$. In this step, we use common neighborhood information to form the initial communities. Therefore, by considering parameter k as the node degree average of the given network, the time complexity of this step will be $O(n'k)$. The second step is *updating the labels*, which according to Equation (1), each neighbor updates its label once. This step is also performed for the n' node of the previous step, which calculates the common neighbor and then performs the update procedure which in sum yields to $O(n'k)$. In this step, the label of degree 1 nodes must be assigned based on their respective neighbors, and since

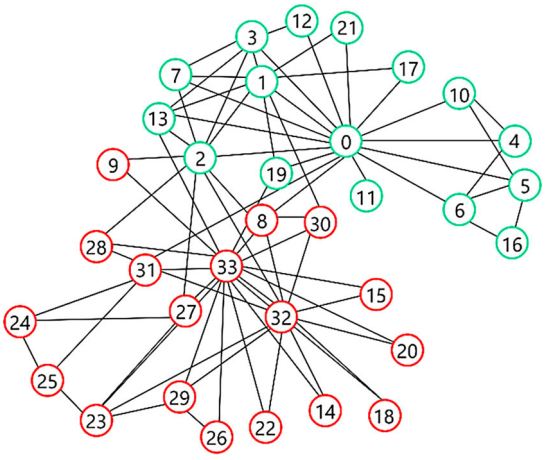

Figure 9. Diffuse the label to node 11 and form the final communities.

the number of nodes with degree 1 is $n - n'$, this task will require the time complexity $O(n - n')$. Therefore, the total time complexity of this step will be $O(n - n' + n'k)$. The last step is to *merge* the small and weak communities with one of the neighboring communities. For S small community with L nodes, first the size of each community and the average size of the community in the network are calculated $O(L)$. Communities smaller than the average size will be considered for merging. Thus, checking merge condition for the S small community has the time complexity $O(Lk)$. As a result, in sum, the total time complexity of the proposed algorithm will be $O(n'k + n - n' + n'k + Lk)$. Since $Lk < nk$ and $n'k < nk$, then the total computational complexity of the proposed algorithm will be $O(n'k)$ that shows FSLD algorithm does not use heavy computation in community detection. It is definitely mentioned that this algorithm is one of the best methods for community detection in terms of simplicity, execution time efficiency, and accuracy.

4. Experiments

Our experiments have been conducted on several real-world and synthetic networks. The proposed algorithm has been compared with some of the recently presented algorithms such as CNM (Clauset, Newman, and Moore 2004), LPA (Raghavan, Albert, and Kumara 2007), Infomap (Rosvall and Bergstrom 2008), Louvain (Blondel et al. 2008), NIBLPA (Xing et al. 2014), Stepping LPA-S (Li et al. 2017), LCCD (Wang et al. 2017), ECES (Berahmand, Bouyer, and Vasighi 2018), LPA-Intimacy (Kong et al. 2018), RTLCD (Ding, Zhang, and Yang 2018), G-CN (Tasgin and Bingol 2019), Leiden (Traag, Waltman, and Van Eck 2019), CFCD2 (Zhang, Ding, and Yang 2019), LSMD (Bouyer and Roghani 2020), and LCDR (Aghaalizadeh et al. 2021). All the experiments were performed on a computer with a core i5 processor (1.60–1.80 GHz) and 8 GB of memory in PyCharm IDE (2019) and Windows 10 operating system. Furthermore, for two large-scale networks (LiveJournal and Orkut networks), we have used other computer system with 12GB RAM and Core i7 CPU.

Table 2. The properties of real-world networks.

Network	Nodes	Edges	Number of communities
Zachary's Karate club (Zachary 1977)	34	78	2
Dolphins (Lusseau et al. 2003)	62	159	2
Football (Girvan and Newman 2002)	115	613	12
Zachary's Karate club (Zachary 1977)	105	441	3
Dolphins (Lusseau et al. 2003)	334863	925872	75149
DBLP (Yang and Leskovec 2015)	317181	1149866	13447
YouTube (Yang and Leskovec 2015)	1134891	2987624	8385
LiveJournal (Yang and Leskovec 2015)	3997962	34681189	287512
Orkut (Yang and Leskovec 2015)	3072441	117185083	6288363
Email (Guimera et al. 2003)	1133	5451	Unknown
Email-Enron (Leskovec et al. 2009)	36692	183831	Unknown
Condmat2003 (Newman 2004)	31163	121129	Unknown
Condmat2005 (Newman 2001)	41421	175691	Unknown
Power grid (Watts and Strogatz 1998)	4941	6594	Unknown
NetScience (Newman 2006)	1589	2742	Unknown
PGP (Boguná et al. 2004)	10680	24316	Unknown

4.1. Real-world data sets

To evaluate the proposed FSLD algorithm, it is primarily tested in real-world data sets to compare its performance with other state-of-the-art algorithms. Therefore, sixteen real-world well-known networks including small, medium, and large-scale networks. The smallest data set is Zachary's Karate club data set with 34 nodes, 78 edges, and 2 communities. The largest data set is LiveJournal data set with 3,997,962 nodes, 34,681,189 edges, and 287,512 communities. Detailed information about the data sets is shown in Table 2. In these data sets, N, M, CN, and k represent the number of nodes, edges, number of communities, and the average degree of network, respectively.

4.2. Synthetic networks

Synthetic networks with adjustable parameters are used to test the FSLD method. LFR (Lancichinetti, Fortunato, and Radicchi 2008) is one of the most popular methods of producing synthetic networks that show realistic features similar to the real-world networks due to the control of node degree distribution and community size distribution. Correspondingly, LFR has adjustable parameters to adjust the number of nodes, community, the distribution of degrees, and the number of community members. One of the important parameters of LFR is the mixing parameter (μ), which indicates the probability of node connections to other nodes between communities. Some of the important parameters are shown in Table 3, and the details of the generated data sets are shown in Table 4.

4.3. Evaluation parameters and measures

NMI, F-measure and modularity measures are used to evaluate the performance of the proposed algorithm in the real-world and LFR synthetic datasets. NMI (Danon et al. 2005) is a measure for evaluating the results of the detected communities by the algorithm with real communities. It only is applicable in networks with ground truth. The NMI measure

Table 3. The parameters of LFR synthetic networks.

Parameter	Description
N	Number of nodes
Min K	Minimum degree of nodes
Max K	Maximum degree of nodes
μ	Mixing parameter
Min c	Number of nodes within the smallest community
Max c	Number of nodes within the biggest community
γ	Node degree distribution exponent
β	Community size distribution exponent

Table 4. The properties of LFR synthetic networks.

Network	N	Mink	Maxk	Minc	Maxc	β	γ	μ
LFR1	5000	20	50	10	50	1	2	0.1–0.8
LFR2	5000	20	50	20	100	1	2	0.1–0.8
LFR3	10000	20	50	10	50	1	2	0.1–0.8
LFR4	10000	20	50	20	100	1	2	0.1–0.8
LFR5	20000	20	50	10	50	1	2	0.1–0.8
LFR6	20000	20	50	20	100	1	2	0.1–0.8
LFR7	50000	20	50	10	50	1	2	0.1–0.8
LFR8	50000	20	50	20	100	1	2	0.1–0.8
LFR9	100000	20	50	10	50	1	2	0.1–0.8
LFR10	100000	20	50	20	100	1	2	0.1–0.8

is defined according to Equation (3):

$$NMI(A,B) = \frac{-2\sum_{i=1}^{C_A}\sum_{j=1}^{C_B} N_{ij}\log\left(\frac{N_{ij}N}{N_i N_j}\right)}{\sum_{i=1}^{C_A} N_i \log\left(\frac{N_i}{N}\right) + \sum_{j=1}^{C_B} N_j \log\left(\frac{N_j}{N}\right)} \tag{3}$$

In Equation (3), $NMI(A,B)$ shows the similarity between partitions based on the information theory. A represents the real communities and B represents the detected communities by the algorithm. Also, the number of actual communities and the number of detected communities are denoted by C_A and C_B, respectively. N is the confusion matrix where the number of rows and columns are equal to the number of nodes in the network. N_{ij} is equal to the number of common nodes between the real community i with the detected community j; N_i and N_j are equal to the sum of row i in the matrix N_{ij} and the sum of column j in the matrix N_{ij}, respectively. If the discovered communities by the algorithm are the same as the real communities, $NMI(A,B)$ will have a maximum value of 1. If the discovered communities by the algorithm are completely independent of the real communities, e. g., when the whole network is identified as a single community, $NMI(A,B) = 0$. F-measure is used to evaluate the accuracy of algorithms. This measure is defined as the harmonic mean between precision and recall measures (Chinchor 1992). F-measure balances the precision and recall values and its value is between 0 and 1. If the precision and recall of the evaluated method are equal to 1, F-measure will have the highest value, which indicates the high accuracy of the algorithm. Using this measure, the performance of the algorithm in finding the correct and incorrect answers is evaluated. Precision, recall, and F-measure are defined by Equations (4)–(6), respectively.

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

$$F - measure = 2 \times \frac{precision \times recall}{precision + recall} \tag{6}$$

In Equations (4) and (5), *TP* denotes *true-positive* results. The positive results that the algorithm has correctly identified. *FP* denotes *false-positive* results. The results that the algorithm has not identified them correctly. In fact, the nodes that are in the same community but the algorithm has not detected them in that community. *FN* denotes *false-negative* results, i.e. the nodes that are not in the same community but the algorithm has identified them in the same community.

Modularity is an evaluation measure for discovered communities that are mostly used in networks without ground truth. This measure evaluates the density of community's inner edges relative to the community's outer edges (Clauset, Newman, and Moore 2004). The modularity measure computes the quality of detected communities by comparing the number of edges of a community with the expected value for a random network of the same size and degree. However, this measure is not the exact measure for evaluation. The modularity value is between -1 and 1. The closer this value to 1, the better the quality of partitioning is. The modularity measure is defined according to Equation(7).

$$Q = \frac{1}{2m} \sum_{i,j \in V} \left(A_{ij} - \frac{d_i d_j}{2m} \right) \times \delta(c_i, c_j) \tag{7}$$

In Equation (7), m denotes number of edges in the network, and A is the adjacency matrix of the network. If there is an edge between i and j, $A_{ij} = 1$, otherwise $A_{ij} = 0$. d_i and d_j represent the degrees of node i and node j, respectively. c_i and c_j denotes the labels of node i and node j, respectively. $\delta(c_i,c_j)$ is Kronecker delta and when two nodes have the same label $\delta(c_i,c_j) = 1$, otherwise it is equal to zero. It is vital to mention that in networks with ground truth, NMI and F-measure are two suitable measures for evaluation.

4.4. Results of NMI evaluation on real-world data sets with ground truth

Initially, evaluations are performed on the datasets with ground truth using the NMI measure to show the accuracy of the algorithms. The NMI results for 9 networks with ground truth are shown in Tables 5 and 6. The obtained NMI result on small data sets is represented in Table 5; and the NMI on large-scale networks is demonstrated in Table 6. Small networks include the Karate, Dolphins, Polbooks, and Football data sets. The large-scale networks include the DBLP, Amazon, YouTube, LiveJournal, and Orkut.

Table 5 shows the evaluation results of the algorithms using the NMI measure in small networks. The obtained results show that the FSLD, LCDR, and LSMD algorithms have the best performance in the Karate and Dolphins data sets. These methods have correctly identified the communities without any mistakes with NMI $= 1$. In the Polbooks and Football data sets, the highest value of NMI belongs to the LSMD algorithm. In these two data sets, the FSLD algorithm has just obtained better results than some compared algorithms.

Table 6 shows the evaluation results of the algorithms using the NMI measure in large-scale networks. Unlike small data sets, the proposed FSLD algorithm meaningfully

Table 5. NMI results obtained from experiments on small real-world networks with ground truth.

Data set algorithm	Karate	Dolphins	Polbooks	Football
CNM	0.69	0.55	0.53	0.75
Infomap	0.7	0.54	0.5	0.91
LCCD	0.71	0.53	0.52	0.88
ECES (s = 1)	0.63	0.49	0.5	0.75
LPA	0.68	0.53	0.52	0.81
NIBLPA	0.58	0.5	0.53	0.72
LPA-Intimacy	1	0.63	0.54	0.86
Stepping LPA-S	0.92	0.88	0.57	0.92
RTLCD	1	0.45	0.48	0.51
Louvain	0.59	0.52	0.45	0.89
Leiden	0.687	0.55	0.57	0.85
CFCD2	0.84	0.55	0.48	0.83
G-CN	0.83	0.54	0.53	0.87
LSMD	1	1	0.59	0.93
LCDR	1	1	0.58	0.90
Proposed FSLD method	1	1	0.52	0.90

Table 6. NMI results obtained from experiments on large-scale real-world networks with ground truth.

Data set algorithm	DBLP	Amazon	YouTube	LiveJournal	Orkut
CNM	0.16	0.11	N/A	N/A	N/A
Infomap	0.47	0.51	0.13	N/A	N/A
ECES (s = 1)	0.36	0.58	0.17	N/A	N/A
LPA	0.49 ± 4	0.53 ± 5	0.07 ± 1	N/A	N/A
NIBLPA	0.46	0.6	N/A	N/A	N/A
LPA-Intimacy	0.61	0.63	N/A	N/A	N/A
RTLCD	0.41	0.72	N/A	N/A	N/A
Louvain	0.13	0.11	0.06	0.03	0.06
Leiden	0.54	0.86	0.46	N/A	N/A
CFCD2	0.26	0.72	0.30	N/A	N/A
G-CN	0.51	0.59	0.07	0.17	N/A
LSMD	0.50	0.71	0.19	0.80	0.29
LCDR	0.70	0.69	0.31	N/A	N/A
Proposed FSLD method	0.74	0.95	0.51	0.92	0.38

outperforms all compared algorithms in all large-scale networks. The FSLD method in the DBLP data set has obtained NMI = 0.74, which is the highest NMI value among other compared algorithms. In addition in this data set, the LCDR algorithm has the second highest NMI value (NMI = 0.70). The FSLD algorithm in the Amazon and YouTube data sets has NMI values of 0.95 and 0.51, respectively, which are the best results compared with other algorithms. In addition, some methods, such as CNM, RTLCD, NIBLPA, and LPA-Intimacy, failed to run on the YouTube data set after two days in the provided computer system, and their NMI values are unknown (N/A). Furthermore, the proposed FSLD algorithm obtained the NMI value of 0.92 and 0.38 in LiveJournal and Orkut data sets, respectively that significantly outperforms other compared algorithms. Except LSMD and Louvain methods, other algorithms cannot be executed on LiveJournal and Orkut data sets. In sum, the proposed algorithm has the highest value of NMI in all large-scale networks, which shows this algorithm outperforms other methods in terms of accuracy in the small and large-scale data sets.

Table 7. Average F-measure results obtained from the experiments on real-world data sets with ground truth.

Dataset Algorithm	Karate	Dolphins	Polbooks	Football	DBLP	Amazon	YouTube
CNM	0.54	0.41	0.52	0.38	0.23	0.28	N/A
Infomap	0.59	0.33	0.43	0.81	0.34	0.39	0.15
ECES (s = 1)	0.55	0.39	0.51	0.63	0.33	0.37	N/A
LPA	0.58	0.29	0.55	0.6	0.33	0.66	N/A
NIBLPA	0.39	0.39	0.53	0.47	0.4	0.66	N/A
LPA-Intimacy	1	0.43	0.61	0.68	0.45	0.71	N/A
RTLCD	1	0.73	0.66	0.6	0.37	0.74	N/A
Louvain	0.5	0.3	0.46	0.74	0.34	0.51	0.29
Leiden	0.81	0.64	0.61	0.87	0.15	0.27	0.1
G-CN	0.95	0.29	0.38	0.76	0.36	0.69	0.11
LSMD	1	1	0.7	0.91	**0.49**	0.82	0.13
Proposed FSLD method	**1**	**1**	**0.97**	**0.92**	0.47	**0.84**	**0.82**

4.5. Results of F-measure evaluation on the real-world data sets with ground truth

F-measure is an accurate measure for evaluating the accuracy of detected communities. The results obtained by the F-measure criterion are shown in Table 7. The obtained results show the ability of the algorithm to determine the correct and incorrect results. Since the F-measure is calculated for each class or community, to compare the algorithms, we calculate the average F-measure of each algorithm on a specific data set.

In the Karate data set, the F-measure value for the FSLD, RTLCD, LPA-Intimacy and LSMD is 1 that is the best possible F-measure value. On the Dolphins data set, the proposed algorithm and the LSMD have F-measure equal to 1 and have correctly identified all the communities. In this data set, the second highest value of F-measure belongs to the RTLCD algorithm but it significantly is lower than the F-measure of FSLD algorithm. In the Polbooks dataset, the FSLD algorithm has obtained the best result compared with other algorithms (F-measure = 0.97). On the Football data set, the proposed algorithm has the best F-measure result equal to 0.92, and the second rank F-measure is for the LSMD algorithm. In the DBLP data set, the LSMD algorithm obtained the highest result with F-measure = 0.49 and FSLD with F-measure = 0.47 are in second rank. In the Amazon data set, the FLSD algorithm with F-measure = 0.84 has the highest result; and the second rank is for the LSMD algorithm with F-measure = 0.82. On the YouTube data set, the proposed algorithm with F-measure = 0.82 is the best method among compared algorithms. Analysis of these tables reveals that the proposed method has the best results in all datasets with ground truth, except in DBLP, compared to other algorithms. The results also show that the proposed algorithm finds nearly real communities, the actual number of communities and more accurate results on these datasets.

4.6. Results of modularity evaluation on the real-world data sets

In addition to the NMI and F-measure criteria that are used to evaluate experiment results, the modularity criterion is used when the network does not have ground truth. This measure is used to evaluate the quality of the detected communities. Table 8 shows the modularity and number of detected communities by the proposed algorithm and other algorithms in the real-world data sets.

Table 8. Modularity results obtained from the experiments on real-world data sets.

	Louvain		CNM		LPA		LCDR		Infomap		G-CN		LSMD		Proposed FSLD		Ground- truth	
Network	CN	Q	CN	Q	CN	Q	CN	Q	CN	Q	CN	Q	CN	Q	CN	Q	CN	Q
Karate[a]	4	0.41	3	0.38	3	0.38	**2**	**0.371**	3	0.40	**2**	**0.371**	**2**	**0.371**	**2**	**0.371**	2	0.371
Dolphins[a]	6	0.51	4	0.49	5	0.46	**2**	**0.378**	5	0.53	5	0.51	**2**	**0.378**	**2**	**0.378**	2	0.378
Polbooks[a]	5	0.52	4	0.50	4	0.50	5	0.502	5	0.52	6	0.51	3	0.446	3	**0.445**	3	0.4149
Football[a]	10	0.60	6	0.54	9	0.54	14	0.602	12	0.58	12	**0.57**	12	0.584	8	0.495	12	0.554
DBLP[a]	225	0.81	3301	0.72	30384	0.60	19405	0.693	16921	0.71	22056	0.71	17280	0.65	**14463**	**0.62**	13477	0.61
Amazon[a]	286	0.90	6086	0.87	27959	0.64	21749	0.778	17319	0.75	29484	0.759	34304	0.68	**49128**	**0.73**	75149	0.74
Email	15	0.54	17	0.48	77	0.34	16	**0.543**	60	0.52	624	0.35	142	0.42	34	0.37	Unknown	Unknown
Email_ Enron	1463	0.55	1605	0.51	2112	0.38	1612	**0.56**	2605	0.52	1751	0.14	1221	0.39	2149	0.49	Unknown	Unknown
NetScience	276	0.92	276	0.90	336	0.91	332	0.92	294	0.90	317	0.88	275	**0.95**	473	0.81	Unknown	Unknown
Power_ Grid	57	0.84	41	**0.89**	774	0.74	473	0.79	496	0.79	678	0.64	446	0.79	415	0.73	Unknown	Unknown
Condmat_ 2003	98	**0.72**	1916	0.66	3814	0.53	3122	0.681	2540	0.61	3387	0.61	3502	0.57	2068	0.63	Unknown	Unknown
Condmat_ 2005	1077	**0.71**	2253	0.63	4503	0.50	3691	0.645	2966	0.63	3906	0.62	3952	0.44	2177	0.60	Unknown	Unknown
PGP	96	**0.88**	191	0.85	2059	0.74	869	0.80	3	0.13	911	0.79	643	0.59	302	**0.88**	Unknown	Unknown

[a]Networks with ground truth (the bold values show the better modularity values).

The analysis of Table 8 reveals that in the Karate data set, the proposed algorithm identifies the communities correctly without any mistakes and obtains a modularity value of 0.371, which is equal to the actual modularity whereas some methods such as Louvain, CNM, and Infomap have higher modularity value than 0.371 that necessarily is not better value because the values of NMI and F-measure for these methods are lower than 1. It is revealed that modularity is not exact measure for evaluation. To show the effectiveness of compared algorithm, we compared the obtained modularity value with real modularity in data sets with ground truth. Therefore, the analysis of FSLD and other compared algorithms show that the FSLD method has the closest modularity value to real modularity in all nine data sets with ground truth, except football data set.

On the other hand, the proposed algorithm is evaluated based on the modularity in several data sets without ground truth. In some of these data sets, the proposed algorithm has gained the higher modularity. However, in these networks, Louvain has obtained the highest modularity. In general, all modularity optimization-based algorithms mostly have higher modularity than others but they have lowest accuracy in comparison to other methods. For example, analysis of datasets with ground truth shows that most of compared algorithms do not have better NMI and F-measure results but they have higher modularity.

4.7. The experiment results on LFR synthetic datasets

A number of experiments were conducted on 10 LFR synthetic datasets with different parameters μ ranging from 0.1 to 0.8. Increasing the value of μ decreases the resolution of the network, in which case there are more connections between the communities in the network. When the values of μ are lower, the community detection algorithms may produce similar results. However, when the value of μ increases, algorithms that are more stable can provide better results. In this section, the results of the experiments are measured using the NMI criterion. Figures 10–19 show a comparison of NMI results for 5 algorithms on 10 types of the synthetic data sets.

Figure 10 shows the NMI experiment results of the LPA, LPA-Intimacy, NIBLPA, G-CN algorithms, and the proposed algorithm on the LFR1 data set with 5000 nodes. According to this figure, when $\mu < 0.4$, almost all algorithms have achieved good results, and the proposed FLSD, LSMD, and G-CN algorithm have better results than other methods. In general, for $0.1 \leq \mu \leq 0.7$, FSLD and G-CN algorithms have the highest values of NMI. In $\mu = 0.5$, the proposed FSLD algorithm outperforms all other compared methods. In addition, the three algorithms LPA, LPA-Intimacy, and NIBLPA significantly have lower NMI value for $\mu > 0.4$. Furthermore, for $\mu = 0.8$, the accuracy of these algorithms is zero, and they are not able to correctly detect communities.

In Figure 11, when $\mu < 0.6$, the proposed algorithm has higher NMI value than other algorithms. With increasing the μ value, LPA, LPA-Intimacy, and NIBLPA algorithms do not correctly identify communities, and their NMI values are reduced to zero. However, the proposed algorithm has obtained an acceptable result for high values of μ. In $\mu \geq 0.7$, the G-CN algorithm outperform all other compared methods.

Figures 12 and 13 show the NMI results obtained on data sets with 10000 nodes. In Figure 12, when $0.1 \leq \mu \leq 0.8$, the proposed FSLD algorithm has better NMI value than other methods. Particularly, it is the best for $\mu \geq 0.4$; and its performance significantly higher than the three algorithms LPA, LPA-Intimacy, and NIBLPA.

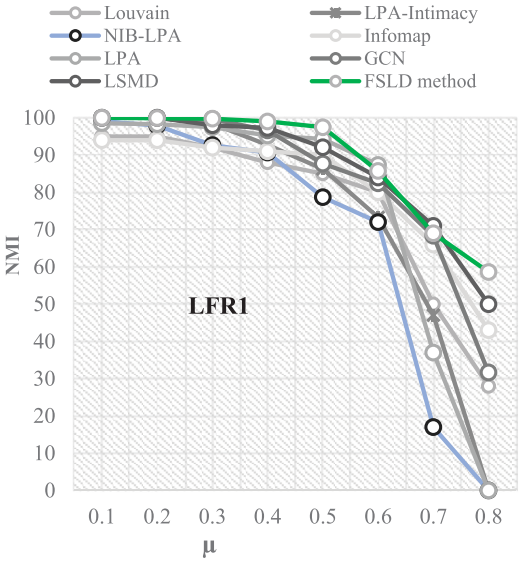

Figure 10. NMI results obtained by LPA, LPA-Intimacy, NIBLP, G-CN, and the proposed method on LFR1 data set.

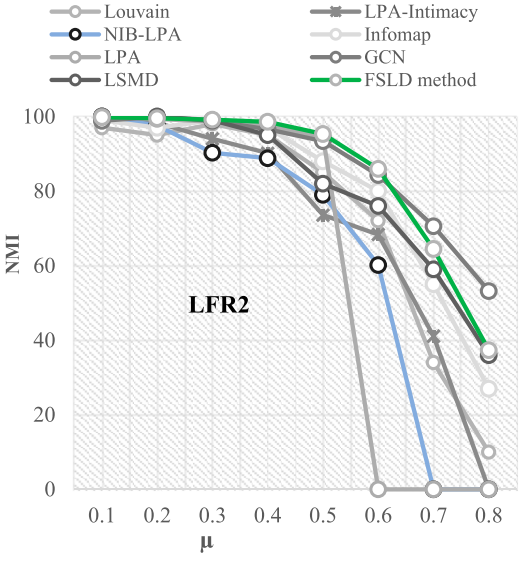

Figure 11. NMI results obtained by LPA, LPA-Intimacy, NIBLP, G-CN, and the proposed method on LFR2 data set.

In Figure 13, when $\mu < 0.4$, most algorithms have higher NMI values. However, with increasing the value of μ, LPA and NIBLPA algorithms decrease significantly and their NMI values are reduced to zero. FSLD method has gained the best NMI for $0.4 \leq \mu \leq 0.6$. However, G-CN is better than FSLD in $0.7 \leq \mu$.

In Figure 14, the FSLD and G-CN algorithms has better than others. When $\mu = 0.8$, G-CN algorithm have better results than other algorithms.

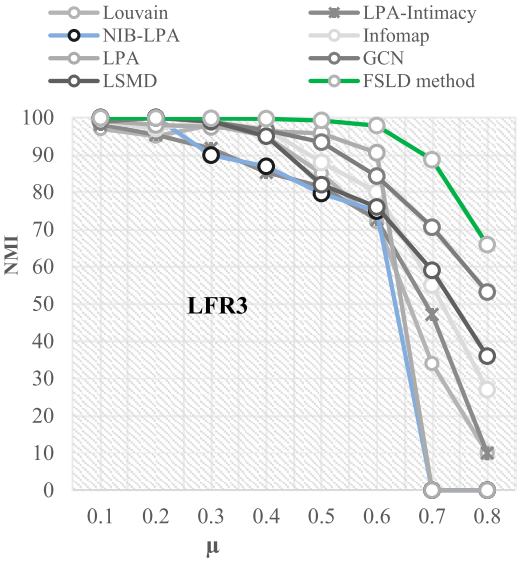

Figure 12. NMI results obtained by LPA, LPA-Intimacy, NIBLP, G-CN, and the proposed method on LFR3 data set.

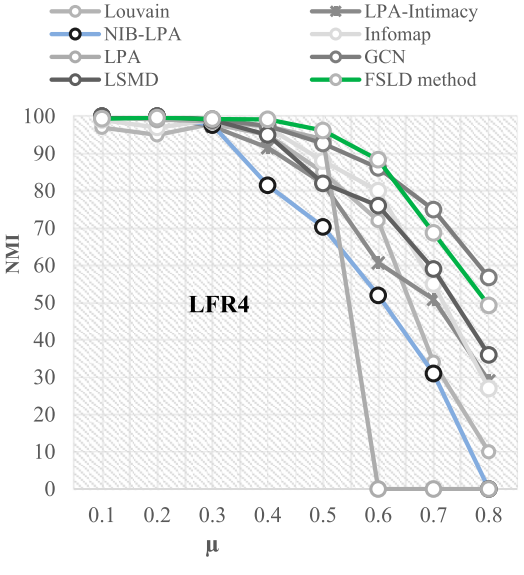

Figure 13. NMI results obtained by LPA, LPA-Intimacy, NIBLP, G-CN, and the proposed method on LFR4 data set.

In Figure 15, when $\mu \leq 0.4$, the FSLD, LSMD and G-CN algorithms have good NMI results. For $0.5 \leq \mu \leq 0.6$, FSLD has obtained best NMI values and LPA algorithm has a significant reduction, but for higher values of μ, the LPA-Intimacy and NIBLPA algorithms have better NMI values than LPA algorithm. However, the proposed algorithm averagely is the best algorithm in LFR 6 network.

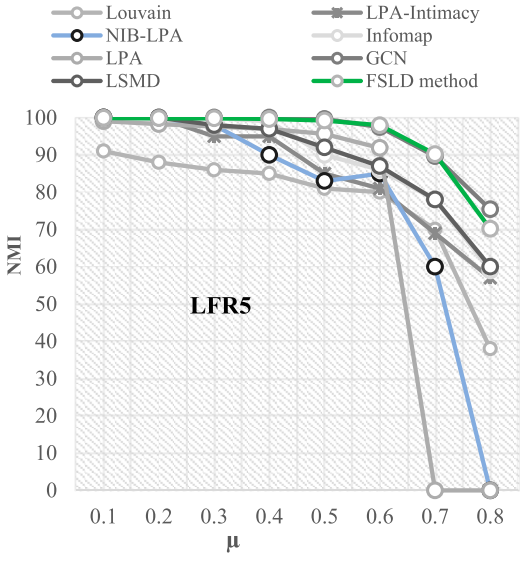

Figure 14. NMI results obtained by LPA, LPA-Intimacy, NIBLP, G-CN, and the proposed method on LFR5 data set.

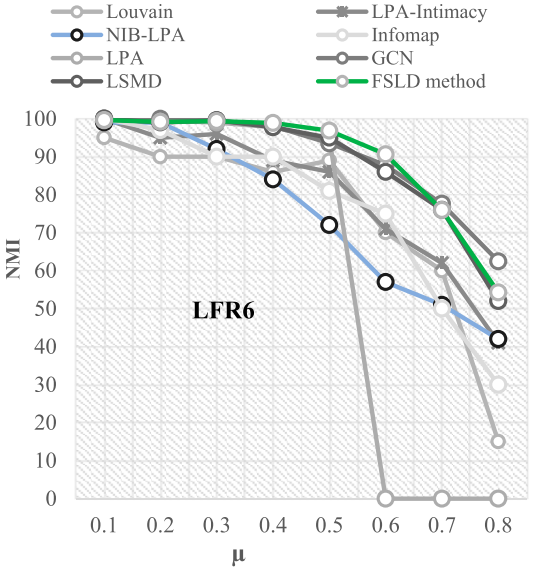

Figure 15. NMI results obtained by LPA, LPA-Intimacy, NIBLPA, G-CN, and the proposed method on LFR6 data set.

In Figure 16, for $\mu < 0.5$, the FSLD, LSMD and G-CN algorithms have better results than other algorithms. For $0.5 \leq \mu \leq 0.8$, the proposed FSLD algorithm has gained the best NMI values, whereas LPA, LPA-Intimacy, and NIBLPA algorithms have obtained lower results than others.

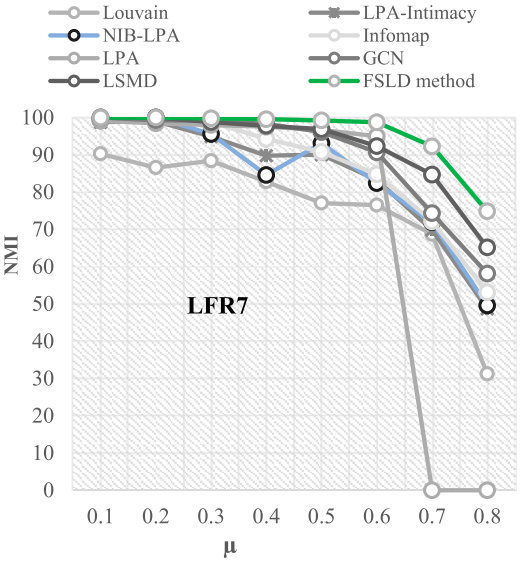

Figure 16. NMI results obtained by LPA, LPA-Intimacy, NIBLP, G-CN, and the proposed method on LFR7 data set.

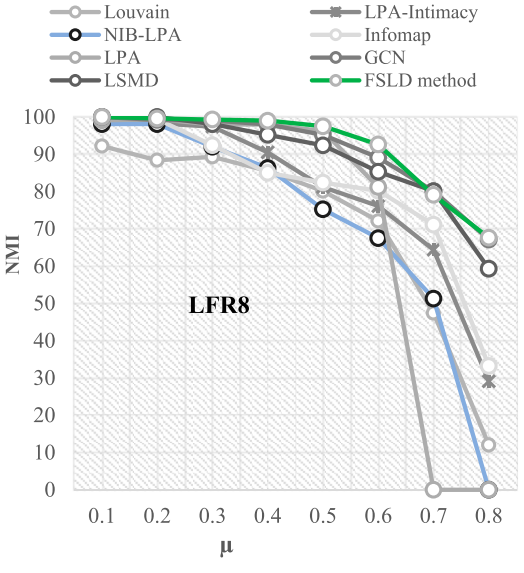

Figure 17. NMI results obtained by LPA, LPA-Intimacy, NIBLP, G-CN, and the proposed method on LFR8 data set.

In Figure 17, for $0.1 \leq \mu \leq 0.8$, the proposed algorithm has better NMI results than other algorithms. However, for $0.6 \leq \mu$, LPA, LPA-intimacy, and NIBLPA algorithms do not perform well and do not correctly identify the communities.

In Figure 18, for $0.1 \leq \mu \leq 0.7$, the FSLD and G-CN algorithms have better NMI results than other algorithms. However, for $\mu = 0.5$ and $\mu = 0.8$, the FSLD and G-CN have obtained best results, respectively.

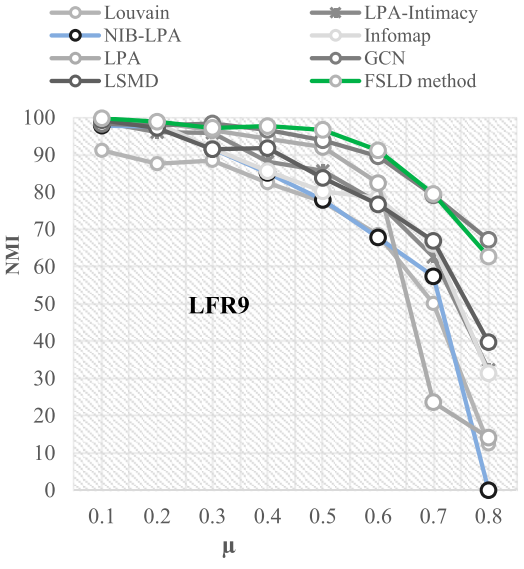

Figure 18. NMI results obtained by LPA, LPA-Intimacy, NIBLP, G-CN, and the proposed method on LFR9 data set.

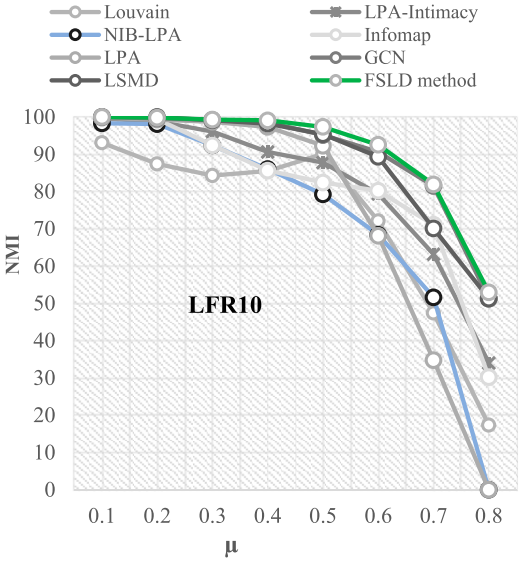

Figure 19. NMI results obtained by LPA, LPA-Intimacy, NIBLP, G-CN, and the proposed method on LFR10 data set.

In Figure 19, for $\mu < 0.5$, the proposed method, LSMD, and G-CN, and LPA algorithms have better results than other algorithms. For $0.5 \leq \mu \leq 0.8$, the proposed algorithm have higher NMI results than other algorithms, and LPA and NIB-LPA have severe reduction.

Consequently, as the size of the network increases, the accuracy of the detection algorithms becomes more different. A more stable algorithm is required to correctly identify the communities in the networks with large size and low resolution. The analysis of Figures

Table 9. Computing the execution times on different networks.

Algorithm dataset	Infomap	Louvain	LPA	NIBLPA	LPA-Intimacy	G-CN	LSMD	Proposed method
PGP	19	15	25	23	29	10	7	2
LFR5	156	77	169	156	182	60	29	11
LFR6	170	76	193	211	193	63	30	14
Condmat2003	169	62	183	206	233	33	22	16
Email-Enron	439	100	340	285	224	77	39	15
Condmat2005	301	90	345	301	387	75	38	22
LFR9	3813	305	854	982	1111	141	92	39
LFR10	3933	291	1061	1096	1145	144	93	35
DBLP	13320	1076	20619	22152	21261	348	179	12
Amazon	9375	853	22735	23102	20134	364	155	10
YouTube	N/A	2993	71030	N/A	N/A	13221	605	223
LiveJournal	N/A	N/A	N/A	N/A	N/A	N/A	7058	1132
Orkut	N/A	N/A	N/A	N/A	N/A	N/A	30286	11387

16–19 reveals that the FSLD algorithm has the best stability with increasing μ. Likewise, G-CN algorithm has higher values of NMI, and similar to FSLD method, it has small reduction in NMI results by increasing values of μ. The results totally show that the proposed FSLD algorithm are more accurate and stable than other compared algorithms.

4.8. Evaluation of execution time

One of the effective criteria in determining the performance of an algorithm is the execution time of the algorithm. An algorithm that has high accuracy must have an acceptable execution time. As mentioned, the proposed FSLD algorithm uses a local method to diffuse the label which reduces greatly the execution time of the algorithm. Table 9 shows the average execution time of the algorithms on the given networks for two run. The displayed execution time of the algorithms is in second unit. In addition, the execution time of LFR networks has been obtained by calculating the average execution time for the eight different mixing parameters. It should be noted that the execution time is calculated only on medium and large-scale networks, since the execution time on small networks is negligible.

Table 9 shows that the proposed algorithm has lowest execution time in all data sets. In the second rank, LSMD algorithm is the faster method than other compared methods. It should be mention that the FSLD significantly is faster than LSMD especially on Large-scale networks such as YouTube, LiveJournal, and Orkut. In other words, most differences in the execution time of the FSLD algorithm with other methods can be observed in the large networks such as DBLP, Amazon, YouTube, LiveJournal, and Orkut. Owing to diffusion label step, the execution time of the proposed algorithm is low. Using this step, the labels of a number of nodes are selected and updated together. In fact, a number of nodes are labeled without time-consuming checks, which has more impact on the accuracy and execution time of the algorithm. The LPA algorithm in most cases has lower execution time than the NIBLPA and LPA-Intimacy algorithms.

5. Conclusion

In this article, a label diffusion method using local criteria with low time complexity and high accuracy was presented to identify communities in complex networks. In the

proposed method, first, nodes with degree two and higher degrees are labeled, and these labels are distributed to the neighbor nodes. Then, in the *update* phase, the labels of degree 2 and higher degree nodes are updated to their neighbor node labels using a local criterion. Eventually, nodes with degree 1 are labeled to their direct neighbor labels and the initial communities are formed. Then, in two consecutive iterations, a *merge* step is used to merge the initial communities which yields the final communities. NMI, F-measure, and modularity criteria were used to evaluate the accuracy of the proposed algorithm. The results of the experiments show the accuracy and better performance of the proposed algorithm in comparison with other community detection algorithms. The proposed algorithm has high accuracy, highest stability without any random behavior, and fast execution speed that made it suitable for detecting communities in large-scale complex networks.

Future researches can improve the updating step of the algorithm by presenting newer local similarity in the label diffusion process instead of using Equation (1). Furthermore, with a small modification on label updating step, it can also be applicable in overlapping community detection.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Notes on contributors

Asgarali Bouyer is an Associate Professor in the Faculty of Computer engineering and information technology at Azarbaijan Shahid Madani University. He received the Ph.D degree in Computer Science from Universiti Teknologi Malaysia (UTM) in 2011. His current research interests are in complex network, social media mining and data clustering, and resource management systems in Cloud/ Grid computing.

Khatereh Azad received the MSc in Information Technology Engineering at Faculty of Computer Engineering and Information Technology at Azarbaijan Shahid Madani University. Her current research interests are in graph clustering and social networks analysis.

Alireza Rouhi is an assistant Professor in the Faculty of Computer engineering and information technology at Azarbaijan Shahid Madani University. He received the Ph.D degree in software engineering from Isfahan University in 2016. His current research interests are in Formal modeling, Design patterns, and machine learning.

ORCID

Alireza Rouhi http://orcid.org/0000-0003-1494-3467

References

Aghaalizadeh, S., Saeid Taghavi Afshord, Asgarali Bouyer, Babak Anari. 2021. "A Three-Stage Algorithm for Local Community Detection Based on the High Node Importance Ranking in Social Networks." *Physica A: Statistical Mechanics and its Applications* 563: 125420.

Barber, M. J., and J. W. Clark. 2009. "Detecting Network Communities by Propagating Labels Under Constraints." *Physical Review E* 80 (2): 026129.

Barnes, E. R. 1982. "An Algorithm for Partitioning the Nodes of a Graph." *SIAM Journal on Algebraic Discrete Methods* 3 (4): 541–550.

Berahmand, K., and A. Bouyer. 2018. "LP-LPA: A Link Influence-Based Label Propagation Algorithm for Discovering Community Structures in Networks." *International Journal of Modern Physics B* 32 (06): 1850062.

Berahmand, K., and A. Bouyer. 2019. "A Link-Based Similarity for Improving Community Detection Based on Label Propagation Algorithm." *Journal of Systems Science Complexity* 32 (3): 737–758.

Berahmand, K., A. Bouyer, and M. Vasighi. 2018. "Community Detection in Complex Networks by Detecting and Expanding Core Nodes Through Extended Local Similarity of Nodes." *IEEE Transactions on Computational Social Systems* 5 (4): 1021–1033.

Blondel, V. D., Jean-Loup Guillaume, Renaud Lambiotte, Etienne Lefebvre. 2008. "Fast Unfolding of Communities in Large Networks." *Journal of Statistical Mechanics: Theory Experiment* 2008 (10): P10008.

Boccaletti, S., V. Latora, Y. Moreno, M. Chavez, D. Hwang. 2006. "Complex Networks: Structure and Dynamics." *Physics Reports* 424 (4–5): 175–308.

Boguná, M., Romualdo Pastor-Satorras, Albert Díaz-Guilera, Alex Arenas. 2004. "Models of Social Networks Based on Social Distance Attachment." *Physical Review E* 70 (5): 056122.

Bouyer, A., and H. Roghani. 2020. "LSMD: A Fast and Robust Local Community Detection Starting from Low Degree Nodes in Social Networks." *Future Generation Computer Systems* 113: 41–57.

Chinchor, N. 1992. "MUC-4 Evaluation Metrics." Proceedings of the 4th conference on message understanding. Association for Computational Linguistics.

Clauset, A., M. E. Newman, and C. Moore. 2004. "Finding Community Structure in Very Large Networks." *Physical Review E* 70 (6): 066111.

Cui, Y., and X. Wang. 2014. "Uncovering Overlapping Community Structures by the Key Bi-Community and Intimate Degree in Bipartite Networks." *Physica A: Statistical Mechanics and its Applications* 407: 7–14.

Cui, Y., and X. Wang. 2016. "Detecting One-Mode Communities in Bipartite Networks by Bipartite Clustering Triangular." *Physica A: Statistical Mechanics its Applications* 457: 307–315.

Cui, Y., X. Wang, and J. Eustace. 2014. "Detecting Community Structure via the Maximal Sub-Graphs and Belonging Degrees in Complex Networks." *Physica A: Statistical Mechanics and its Applications* 416: 198–207.

Cui, Y., X. Wang, and J. Li. 2014. "Detecting Overlapping Communities in Networks Using the Maximal Sub-Graph and the Clustering Coefficient." *Physica A: Statistical Mechanics its Applications* 405: 85–91.

Danon, L., Albert Díaz-Guilera, Jordi Duch, Alex Arenas. 2005. "Comparing Community Structure Identification." *Journal of Statistical Mechanics: Theory Experiment* 2005 (9): P09008.

Ding, X., J. Zhang, and J. Yang. 2018. "A Robust Two-Stage Algorithm for Local Community Detection." *Knowledge-Based Systems* 152: 188–199.

Eustace, J., X. Wang, and Y. Cui. 2015a. "Community Detection Using Local Neighborhood in Complex Networks." *Physica A: Statistical Mechanics its Applications* 436: 665–677.

Eustace, J., X. Wang, and Y. Cui. 2015b. "Overlapping Community Detection Using Neighborhood Ratio Matrix." *Physica A: Statistical Mechanics and its Applications* 421: 510–521.

Fortunato, S. 2010. "Community Detection in Graphs." *Physics Reports* 486 (3-5): 75–174.

Fortunato, S., V. Latora, and M. Marchiori. 2004. "Method to Find Community Structures Based on Information Centrality." *Physical Review E* 70 (5): 056104.
Gamgne Domgue, F., N. Tsopze, and R. Ndoundam. 2020. "Community Structure Extraction in Directed Network Using Triads." *International Journal of General Systems* 49 (8): 819–842.
Girvan, M., and M. E. Newman. 2002. "Proceedings of the National Academy of Sciences." Community Structure in Social and Biological Networks.
Guimera, R., L. Danon, A. Díaz-Guilera, F. Giralt, A. Arenas. 2003. "Self-similar Community Structure in a Network of Human Interactions." *Physical Review E* 68 (6): 065103.
Hosseini, R., and A. Rezvanian. 2020. "AntLP: Ant-Based Label Propagation Algorithm for Community Detection in Social Networks." *CAAI Transactions on Intelligence Technology* 5 (1): 34–41.
Kernighan, B. W., and S. Lin. 1970. "An Efficient Heuristic Procedure for Partitioning Graphs." *The Bell System Technical Journal* 49 (2): 291–307.
Kong, H., Qinma Kang, Chao Liu, Wenquan Li, Hong He, Yunfan Kang. 2018. "An Improved Label Propagation Algorithm Based on Node Intimacy for Community Detection in Networks." *International Journal of Modern Physics B* 32 (25): 1850279.
Lancichinetti, A., S. Fortunato, and F. Radicchi. 2008. "Benchmark Graphs for Testing Community Detection Algorithms." *Physical Review E* 78 (4): 046110.
Lei, Y., Y. Zhou, and J. Shi. 2019. "Overlapping Communities Detection of Social Network Based on Hybrid C-Means Clustering Algorithm." *Sustainable Cities Society* 47: 101436.
Leskovec, J., Kevin J. Lang, Anirban Dasgupta, Michael W. Mahoney. 2009. "Community Structure in Large Networks: Natural Cluster Sizes and the Absence of Large Well-Defined Clusters." *Internet Mathematics* 6 (1): 29–123.
Li, C., H. Chen, T. Li, and X. Yang. 2021a. "A Stable Community Detection Approach for Complex Network Based on Density Peak Clustering and Label Propagation." *Applied Intelligence*, 1–21.
Li, W., Ce Huang, Miao Wang, Xi Chen. 2017. "Stepping Community Detection Algorithm Based on Label Propagation and Similarity." *Physica A: Statistical Mechanics its Applications* 472: 145–155.
Li, H., Ruisheng Zhang, Zhili Zhao, Xin Liu. 2021b. "LPA-MNI: An Improved Label Propagation Algorithm Based on Modularity and Node Importance for Community Detection." *Entropy* 23 (5): 497.
Li, X., Shuming Zhou, Jiafei Liu, Guanqin Lian, Gaolin Chen, Chen-Wan Lin. 2019. "Communities Detection in Social Network Based on Local Edge Centrality." *Physica A: Statistical Mechanics its Applications* 531: 121552.
Liu, Z., and Y. Ma. 2019. "A Divide and Agglomerate Algorithm for Community Detection in Social Networks." *Information Sciences* 482: 321–333.
Liu, X., and T. Murata. 2010. "Advanced Modularity-Specialized Label Propagation Algorithm for Detecting Communities in Networks." *Physica A: Statistical Mechanics its Applications* 389 (7): 1493–1500.
Lusseau, D., Karsten Schneider, Oliver J. Boisseau, Patti Haase, Elisabeth Slooten, Steve M. Dawson. 2003. "The Bottlenose Dolphin Community of Doubtful Sound Features a Large Proportion of Long-Lasting Associations." *Behavioral Ecology and Sociobiology* 54 (4): 396–405.
Moradi, F., T. Olovsson, and P. Tsigas. 2014. "A Local Seed Selection Algorithm for Overlapping Community Detection." 2014 IEEE/ACM International conference on advances in Social Networks analysis and mining (ASONAM 2014). IEEE.
Newman, M. E. 2001. "The Structure of Scientific Collaboration Networks." *Proceedings of the National Academy of Sciences* 98 (2): 404–409.
Newman, M. E. 2004. "Fast Algorithm for Detecting Community Structure in Networks." *Physical Review E* 69 (6): 066133.
Newman, M. E. 2006. "Finding Community Structure in Networks Using the Eigenvectors of Matrices." *Physical Review E* 74 (3): 036104.
Newman, M. E., and M. Girvan. 2004. "Finding and Evaluating Community Structure in Networks." *Physical Review E* 69 (2): 026113.

Niwattanakul, S., J. Singthongchai, E. Naenudorn, and S. Wanapu. 2013. "Using of Jaccard Coefficient for Keywords Similarity." In *Proceedings of the International Multiconference of Engineers and Computer Scientists*, 380–384.

Pan, Y., De-Hua Li, Jian-Guo Liu, Jing-Zhang Liang. 2010. "Detecting Community Structure in Complex Networks via Node Similarity." *Physica A: Statistical Mechanics and its Applications* 389 (14): 2849–2857.

Pan, X., Guiqiong Xu, Bing Wang, Tao Zhang. 2019. "A Novel Community Detection Algorithm Based on Local Similarity of Clustering Coefficient in Social Networks." *IEEE Access* 7: 121586–121598.

Raghavan, U. N., R. Albert, and S. Kumara. 2007. "Near Linear Time Algorithm to Detect Community Structures in Large-Scale Networks." *Physical Review E* 76 (3): 036106.

Ren, G., and X. Wang. 2014. "Epidemic Spreading in Time-Varying Community Networks." *Chaos: An Interdisciplinary Journal of Nonlinear Science* 24 (2): 023116.

Roghani, H., A. Bouyer, and E. Nourani. 2021. "PLDLS: A Novel Parallel Label Diffusion and Label Selection-Based Community Detection Algorithm Based on Spark in Social Networks." *Expert Systems with Applications* 183 (2021): 115377.

Rosvall, M., and C. Bergstrom. 2008. "Maps of Random Walks on Complex Networks Reveal Community Structure." *Proceedings of the National Academy of Sciences* 105 (4): 1118–1123.

Salton, G. 1975. *A Theory of Indexing*. Vol. 18. Cornell University.

Taheri, S., and A. Bouyer. 2020. "Community Detection in Social Networks Using Affinity Propagation with Adaptive Similarity Matrix." *Big Data* 8 (3): 189–202.

Tasgin, M., and H. Bingol. 2019. "Community Detection Using Boundary Nodes in Complex Networks." *Physica A: Statistical Mechanics and its Applications* 513: 315–324.

Traag, V. A., L. Waltman, and N. J. Van Eck. 2019. "From Louvain to Leiden: Guaranteeing Well-Connected Communities." *Scientific Reports* 9 (1): 1–12.

Wang, X., and J. Li. 2013. "Detecting Communities by the Core-Vertex and Intimate Degree in Complex Networks." *Physica A: Statistical Mechanics and its Applications* 392 (10): 2555–2563.

Wang, X., G. Liu, J. Li, and J. P. Nees. 2017. "Locating Structural Centers: A Density-Based Clustering Method for Community Detection." *PloS one* 12 (1): e0169355.

Wang, X., and X. Qin. 2016. "Asymmetric Intimacy and Algorithm for Detecting Communities in Bipartite Networks." *Physica A: Statistical Mechanics its Applications* 462: 569–578.

Wang, T., L. Yin, and X. Wang. 2018. "A Community Detection Method Based on Local Similarity and Degree Clustering Information." *Physica A: Statistical Mechanics its Applications* 490: 1344–1354.

Watts, D. J., and S. H. Strogatz. 1998. "Collective Dynamics of 'Small-World'networks." *Nature* 393 (6684): 440.

Xie, J., and B. K. Szymanski. 2012. "Towards Linear Time Overlapping Community Detection in Social Networks." Pacific-Asia conference on Knowledge Discovery and Data Mining, Springer.

Xing, Y., F. Meng, Y. Zhou, M. Zhu, M. Shi, and G. Sun. 2014. "A Node Influence Based Label Propagation Algorithm for Community Detection in Networks." *The Scientific World Journal* 2014: 1–13, Article ID 627581.

Yakoubi, Z., and R. Kanawati. 2014. "LICOD: A Leader-Driven Algorithm for Community Detection in Complex Networks." *Vietnam Journal of Computer Science* 1 (4): 241–256.

Yang, J., and J. Leskovec. 2015. "Defining and Evaluating Network Communities Based on Ground-Truth." *Knowledge and Information Systems* 42 (1): 181–213.

Yin, C., Shuaibing Zhu, Hui Chen, Bingxue Zhang, and Bertrand David. 2015. "A Method for Community Detection of Complex Networks Based on Hierarchical Clustering." *International Journal of Distributed Sensor Networks* 11 (6): 849140.

Zachary, W. W. 1977. "An Information Flow Model for Conflict and Fission in Small Groups." *Journal of Anthropological Research* 33 (4): 452–473.

Zarandi, F. D., and M. K. Rafsanjani. 2018. "Community Detection in Complex Networks Using Structural Similarity." *Physica A: Statistical Mechanics its Applications* 503: 882–891.

Zarezadeh, M., E. Nourani, and A. Bouyer. 2021. "DPNLP: Distance Based Peripheral Nodes Label Propagation Algorithm for Community Detection in Social Networks." *World Wide Web*. Published online 23 November 2021. doi:10.1007/s11280-021-00966-4.

Zhang, J., X. Ding, and J. Yang. 2019. "Revealing the Role of Node Similarity and Community Merging in Community Detection." *Knowledge-Based Systems* 165: 407–419.