Physica A 503 (2018) 882–891

Contents lists available at ScienceDirect

Physica A

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

Community detection in complex networks using structural similarity

Fataneh Dabaghi Zarandi [a,*], Marjan Kuchaki Rafsanjani [b,a]

[a] Department of Computer Science, Faculty of Mathematics and Computer, Shahid Bahonar University of Kerman, Kerman, Iran
[b] Mahani Mathematical Research Center, Shahid Bahonar University of Kerman, Kerman, Iran

HIGHLIGHTS

- We proposed a novel community detection algorithm based on structural similarity.
- It is executed in two phases, community detection and best communities selection.
- It detects communities with high accuracy, is applicable in large & small networks.
- Simulation results show that the proposed algorithm outperforms other algorithms.

ARTICLE INFO

Article history:
Received 12 September 2017
Received in revised form 16 January 2018
Available online 6 March 2018

Keywords:
Community detection
Complex networks
Structural similarity
Modularity

ABSTRACT

These days, community detection is an important field to understand the topology and functions in the complex networks. In this article, we propose a novel Community Detection Algorithm based on Structural Similarity (CDASS) that executed in two consecutive phases. In the first phase, we randomly remove some low similarity edges. Therefore, the network graph is converted into several disconnected components that are considered as primary communities. In the following, the primary communities are merged in order to identify the final community structure close to real communities. In the second phase, we use an our identified evaluation function to select the best communities between overall random generated partitions. Finally, we evaluate CDASS algorithm using several scenarios extracted from artificial and real networks. The results, obtained from simulation with these scenarios, show that proposed algorithm detects communities with high accuracy close to optimal case and is applicable in the large and small network topologies.

© 2018 Published by Elsevier B.V.

1. Introduction

Nowadays, there exist many complex networks such as biological systems, power grids, social networks and any type of communication networks that can be represented by a set of nodes and edges. In this regard, an entity in complex networks can be modeled as a node and connection between two entities is defined as an edge. Commonly, these networks include a big and complex communication graph resulting in difficult management. Therefore, community detection is investigated to overcome this problem that is also an important field for understanding the topology and functions of complex networks. In this regard, a community (or cluster) is defined as a connected subgraph whose its nodes are densely connected with each other. In addition, the nodes included in each community have a sparse connection to the rest of network [1–3]. Consequently, a community is composed of a group of nodes that their similarities are maximal. In this regards, nodes'

* Corresponding author.
E-mail address: dabaghi_fataneh@math.uk.ac.ir (F. Dabaghi Zarandi).

https://doi.org/10.1016/j.physa.2018.02.212
0378-4371/© 2018 Published by Elsevier B.V.

similarity can be computed based on two measures, including: *structural similarity* and *attribute similarity*. The structural similarity is extracted based on network topology that is the most important similarity measure in the community detection. The attribute similarity is computed using the internal characteristics of the individual nodes that is completely independent of the network topology.

As mentioned, the structural similarity measure is extracted using network topology that can be computed by different approaches such as k-distance neighborhood and common neighbors. In k-distance neighborhood approach, two nodes are similar, if the distance (such Manhattan distance) between them is less than k. In common neighbors approach, two nodes are similar, if have more common neighbors, even if they are not adjacent [4–6].

As noted above, the attribute similarity considers the internal characteristics of nodes without any knowledge about network topology and graph structure. For example, the internal attributes of a person node in social networks can include date of birth, sex, affiliation, age and occupation [6–8].

In many community detection approaches, the optimization of quality functions is used that the most famous of them is Newman–Girvan modularity [9]. In the other words, modularity is a measure to evaluate the quality of partitioning in the complex networks that applied in optimization methods to detect communities. In this regards, the modularity value is exchanged based on community structure. In addition, the larger value of modularity indicates the best quality in the detected communities [1,10]. Moreover, the time complexity of modularity maximization in the optimal version of community detection is proved NP-complete [11]. Therefore, the most proposals in this field are included in the greedy or heuristic problem.

In this paper, we introduce a novel Community Detection Algorithm based on Structural Similarity (CDASS) in the complex networks. In addition, we consider unweighted and undirected network topologies. Moreover, CDASS computes similarity of two nodes based on common neighbors measure. Therefore, two nodes are similar, if the number of common neighbors is maximized. Finally, our proposed method extracts the fully non-overlapping communities in each complex network.

The remainder of this article is organized as follows. Section 2 summarizes some related community detection approaches in the complex networks. Section 3 provides basic definitions and related fundamentals. Section 4 explains our proposed method (CDASS), in detail. Section 5 computes time complexity of the proposed method. Section 6 presents experiments and results extracted from several real and artificial scenarios. Finally, Section 7 concludes the article.

2. Related works

In recent years, researchers have investigated many proposals to detect communities in the complex networks. These approaches can be classified in several groups of implementations such as similarity measures, random walk dynamics, label propagation and statistical models. The approaches that use similarity measure, detect community based on the relationship between network nodes and group similar entities in the same community [4,5,12,13]. In random walk approaches, a walker is located on each node and randomly chooses one of the neighbors and exchanges itself position into selected neighbor, in each time step. In this regard, the probability of selecting each neighbor can be increased by a pre-determined probability function [14–16]. Label propagation methods assign a unified label to each node, then the nodes with the same label are considered as a community. In this regards, each node is assigned to an initial label. Afterwards, in each iteration, the label of each node is updated based on the most frequent label of its neighbors until a consistent situation is achieved [17–20]. The approaches that are based on statistical models, the network topology is modeled as statistical methodologies. Then, the communities are extracted based on computational methods. In addition, these approaches are applicable in large networks and networks, including various types of data [21,22]. However, there exist other methods that are not included in this classification such approaches in [1,23,24]. In this article, we introduce a new method (CDASS) and use similarity measures in the community detection process. In the following, we summarize some related proposals in community detection.

In Minimum Spanning Tree and Modularity (MST-M) proposal, network graph is exchanged to a tree including strong similarity edges. Then, $\frac{n-1}{2}$ edges that have high dissimilarity are removed to produce $\frac{n+1}{2}$ disconnect components. Each disconnect component is considered as a primary community. Afterwards, merge operation is executed, iteratively. In merging process, two communities that have more connection (inter-community) than other communities are selected to merge. In each iteration, modularity measure is computed for generated communities and is stored along with its related communities set. Merge operation is repeated until would have existed one community. Finally, the communities with maximum modularity is selected for the best detection [5].

In [4], a community detection approach was proposed that uses hybrid merging of sub-communities method. In this approach, all nodes are assumed unlabeled and each edge in the network graph is assigned by a weight. These weights are computed by common neighbors of two adjacent nodes called cosine similarity. In this regard, edges are sorted based on descending order of their weights. Afterwards, in each iteration, top edge from sorted list is selected and its end nodes form a sub-community. Each node that is assigned to a sub-community is labeled. Therefore, if at least one of nodes connected to selected edge is labeled, that edge can not compose a sub-community and next edge is selected to check. Finally, if there exist some unlabeled nodes, then each of them is considered as a sub-community. Therefore, each sub-community consists of only one or two nodes. In the following, a hybrid approach is used to merge sub-communities achieved to obtain results close to real communities. This approach uses two merging functions including pairwise and single neighbor merging. Pairwise merging approach chooses two sub-community that have maximum modularity and merges them. Single neighbor merging approach merges communities that are connected to only one community, to corresponding community. While, the execution time of finding communities which are connected to only one community is very lower than pairwise.

The modified label propagation method (LPA-S) [19] is performed in two phases. In the first phase, a unique label is assigned to each node and similarity between each pair of nodes are computed. Then, a random visiting order is generated for all nodes in the network topology. The label of each node is updated by the highest similarity value between their neighbors. In this regard, nodes with the same label are included in a subnetwork. Afterwards, in each iteration, the previous steps including generate a random visiting order, update the node label and organize subnetwork are respectively repeated until no changes are made to the node label. Therefore, the result of this phase is producing some subnetworks. In the second phase, the similarity between each pair of subnetworks is computed and a random visiting order is produced for all subnetworks. The label of each subnetwork is updated by the highest value of similarity between their neighbors. In this regard, the subnetworks with the same label are included in a community. Then, in each iteration, the previous steps including generate a random visiting order, update the subnetwork label and organize community are respectively repeated, as long as no changes are made to the subnetwork label. So, the result of this phase is a partitioning on the network topology. Due to using a random order in visiting, steps of this method are repeated to produce various divisions of the network graph. In this regard, the best graph partitioning is selected based on one evaluation function.

3. Background

In this section, we introduce a similarity measure and modularity to find relationship between nodes and evaluate the quality of communities in the complex networks. The similarity measure is used to construct primary communities and the modularity is used to merge primary communities. In order to introduce these definitions, we need to model network topology as a graph that described in the following.

We consider a complex network as an unweighted and undirected graph $G(V, E)$, where $V(G) = \{v_1, v_2, \ldots, v_n\}$ is the set of network nodes and $E(G) = \{(v_i, v_j)|v_i, v_j \in V(G)\}$ is the set of network edges. Each edge determines a connection between a pair of nodes that is represented by (v_i, v_j), $v_i, v_j \in V$. The total number of nodes and edges are defined by $n = |V|$ and $m = |E|$, respectively. In the following sections, we define the similarity measure and modularity based on this definition of network topology.

3.1. Similarity measure

Similarity measure is used to compute the relationship between nodes and determines that nodes can be assigned to the same community. In this article, we define a similarity measure based on common neighbors of two nodes. Therefore, two nodes that have more number of common neighbors are similar. In this regard, the common neighbors of two nodes i and j that can be considered as a structural similarity measure, is computed by Eq. (1).

$$S_{neighbors} = |N_i \cap N_j| \tag{1}$$

where N_i and N_j respectively define the neighbors set of nodes i and j including itself i and j. In the following, normalized similarity of each pair nodes that called cosine similarity is defined in Eq. (2) [25].

$$S_{cosine} = \frac{|N_i \cap N_j|}{\sqrt{|N_i||N_j|}}. \tag{2}$$

3.2. Modularity

In this section, we introduce an optimization of quality function called modularity, which is used in many existing approaches to evaluate detected communities. Therefore, modularity measure can evaluate the quality of graph partitioning and the larger value of modularity represents better quality in the detected communities. The most famous function to compute modularity measure is proposed in [9] that called Newman–Girvan modularity. This function, Q, is computed using Eq. (3).

$$Q = \sum e_{ii} - a_i^2 \tag{3}$$

where e_{ii} is the fraction of edges included in the community i and a_i is the fraction of nodes' degree included in community i. The calculation of e_{ii} and a_i is represented in Eqs. (4) and (5), respectively.

$$e_{ii} = \frac{E_i}{m} \tag{4}$$

where E_i is the number of edges that are included inside of community i and m is the total number of edges in the graph G.

$$a_i^2 = \frac{\sum_{v\in C_i} d_v}{\sum_{v\in G} d_v} \tag{5}$$

where C_i represents community i and d_v is the degree of node v.

In this article, we find small communities at first, and merge these communities in order to obtain results close to real communities. In this regard, communities are merged based on modularity measure to achieve modularity maximization. For this purpose, we use ΔQ that defines difference between modularity after merging two communities compared with primary communities. In addition, ΔQ is also taken from Newman–Girvan modularity and defined in Arab and Afsharchi [4]. In the following, ΔQ is represented by Eq. (6).

$$\Delta Q = \frac{E_{ij}}{m} - 2 \times \frac{2E_i + Ext_i}{2m} \frac{2E_j + Ext_j}{2m} \tag{6}$$

where E_{ij} is the number of edges that are between two independent communities c_i and c_j, Ext_i and Ext_j denote the number of all external edges connected to community c_i and c_j respectively, E_i and E_j are the number of edges that totally included inside of the community c_i and c_j respectively. m is the total edges in the network.

4. Proposed method

As noted above, community detection is an important field to understand the relationship between entities in a graph. In this regard, we introduce a novel Community Detection Algorithm based on Structural Similarity (CDASS) that is performed in two consecutive phases. In the first phase, we randomly remove some low similarity edges result in some disconnect components. Each disconnect component is considered as a primary community. Then, a hybrid approach that explained in Section 2 is used to merge primary communities. To achieve more accuracy, we repeat the first phase several times and save communities set in each iteration. In the second phase, we use an evaluation function that we identify it in this article to select the best communities set among the several communities sets generated in the first phase. In the following, we explain two phases of CDASS method, in detail. These phases are called *community detection* and *best communities selection*, respectively.

4.1. Community detection phase

In this part, we generate several sets of communities based on edges removing. Then, in the second phase of CDASS, which will be explained in Section 4.2, we select the best set between them using our defined measures. To generate these sets, we execute five consecutive steps, including edges weighting, edges removing, primary communities, merging and iteration.

4.1.1. Edges weighting

In the first, we assign a weight to each edge in the network graph. This weight shows the amount of similarity between two adjacent nodes that is computed by Eq. (2) that explained in Section 3.1. After that, the edges of the network are sorted according to their weights in non-decreasing order.

The time complexity of similarity computing is $O(m)$ that proved in [4]. In addition, we use merge–sort algorithm to sort edges that is in $O(m \log m)$ time complexity respectively, where m is the total edges. Therefore, the time complexity of this step is $O(m \log m)$.

4.1.2. Edges removing

In this section, we remove some edges based on sorted list resulted from edge weighting step 4.1.1. Therefore, the edges of the top of the sorted list are removed one by one if their end nodes have degree more than one. Obviously, there exist some edges with the same weight in the sorted list that we select them in a random order to remove. After removing the edges, several disconnected components are generated, which are considered as input of the next step.

In this step, we should check all edges in the sorted list to remove, in worst case. The complexity of each edge checking is $O(1)$, because we have the degree of all end nodes. Therefore, the complexity of this step is calculated based on the length of sorted list that is equal to the number of network edges. Consequently, the edge removing step is included in time complexity of $O(m)$.

4.1.3. Primary communities

Each disconnected component that produced in Section 4.1.2 is considered as a primary community. Every primary community includes at least two nodes, because in edge removing step, we prevent from removing edges that have end nodes with degree one.

The disconnected components can be found according to traversing algorithms of graph such Depth-First Search (DFS) that is executed in time complexity of $O(n)$, where n is the number of network nodes [26]. Therefore, the time complexity of primary communities step is $O(n)$.

4.1.4. Merging

To achieve better communities, we merge primary communities as much as possible. In this regard, we use the hybrid merging approach defined in Arab and Afsharchi [4]. As mentioned in Section 2, this approach uses two merging functions including pairwise and single neighbor merging. The time complexity of merging step is $O(m \log n)$ that is proved in [4].

4.1.5. Iteration

In the previous steps of community detection phase, we achieved a partitioning on the network graph. In Section 4.1.2, we randomly remove some edges. In this regards, we can repeat the step 4.1.2 to 4.1.4, R times to produce several partitioning on the network graph. Finally, the iteration process is stopped when there are no non-repetitive partitioning on the graph.

4.2. Best community selection phase

As mentioned in Section 4.1.5, the process of community detection is repeated to produce several different sets of communities. Therefore, we should select the best set of communities between resulted partitioning from Section 4.1. In this regard, we introduce a new evaluation function to select the best communities that explained in the following.

To determine our identified *evaluation function (EF)*, we need to introduce four definitions, including *number of external edges (NEE)*, *number of internal edges (NIE)*, *internal degree (ID)* and *total degree (TD)*. The *NEE* is defined based on the number of edges in one community that connect that community to another one. The *NIE* is the number of edges that totally included in one community. The *ID* of one community can be calculated based on ($NIE \times 2$). The *TD* of one community is the sum degree of nodes that included in that community. According to these definitions, our identified evaluation function is:

$$EF = \frac{EE}{DD} \tag{7}$$

where *EE* is the ratio of total external edges between communities and sum of internal edges of all communities that is presented in Eq. (8). The community detection is good if the total of intra-community edges ($\frac{1}{2}\sum_{i=1}^{k} NEE(i)$) is high and the total of inter-community edges ($\sum_{i=1}^{k} NIE(i)$) is low. In this regard, the set of community is very good if *EE* is close to zero.

$$EE = \frac{\frac{1}{2}\sum_{i=1}^{k} NEE(i)}{\sum_{i=1}^{k} NIE(i)} \tag{8}$$

where k is the number of communities and *DD* is the ratio to compute the average of fraction total internal degree one community to its total degree. In this regards, partitioning of the network graph is good that the *DD* is high.

$$DD = \frac{1}{k}\sum_{i=1}^{k} \frac{ID(i)}{TD(i)}. \tag{9}$$

As noted above, in Section 4.1.2, we randomly removed edges that produced different partitioning of the graph network. In this regard, we use a novel evaluation function (*EF*) to best chosen set of communities. Therefore, when there is more than one partitioning in a network topology, the lower value of *EF* is the better partitioning of the network graph. The evaluation function is computed based on the number of edges in/out of communities. Therefore, the time complexity of this step is $O(m)$.

4.3. CDASS algorithm

Due to understand the topology and functions of networks, we propose the CDASS algorithm to detect communities in order to classify the network, which simplifies the processing. The Algorithm 1 shows the procedure of community detection by CDASS. The inputs are: $G(V, E)$ is the initial network topology, T is the number of edges for removing, R_m is the number of iterations for merging and F is the fraction of R_m. The output is the community structure with the lowest value of evolution function (*EF*). In the Algorithm 1, the weight of each edge is calculated based on Eq. (2) in the graph G (Line 4). Afterwards, we defined the set of *DC* in order to store detected communities sets (Line 5). In each iteration, we compose one community set on a network graph and save it in set of *DC* until there are no non-repetitive partitioning on the graph (Line 6–23). In this regard, the network edges are sorted based on non-decreasing order of their calculated weights (Line 7). In sorting process, the edges with the same value are listed in random order. Then, we remove T edges that the degree of two their end nodes is not one (Line 10–17). After removing the edges, each disconnect component is considered as a primary community and all of them are stored in set of C (Line 18). In order to produce communities with better modularity, we merge communities included in C using pairwise merging in $\frac{R_m}{F}$ iterations and single neighbor merging in $R_m - \frac{R_m}{F}$ iterations (Line 19–20). After removing communities, we calculate evolution function (*EF*) based on Eq. (7) for resulted communities in C and *EF* and C store in set of *DC* (Line 21–22). Finally, for the best partitioning of network topology, one community structure in *DC* with the lowest value of evolution function (*EF*) is returned as output.

5. Complexity analysis

As mentioned above, the time complexity of step one to step four in community detection phase is $O(m \log m)$, $O(m)$, $O(n)$, $O(m \log n)$ respectively. In addition, the process community detection is repeated R times to produce several various partitioning of the graph. In this regards, steps two to four community detection phase are repeated R times. Therefore, the time complexity of community detection phase is $O(m \log m + R(m + n + m \log n))$. Moreover, the time complexity of best selecting phase is $O(Rm)$. Finally, the complexity CDASS algorithm is $O(m \log m + R(m + n + m \log n + m))$.

Algorithm 1 CDASS algorithm

```
1: procedure COMMUNITY DETECTION
2:     Inputs: $G(V, E)$, the initial network topology; T, the number of edges for removing; $R_m$, the number of
3:             iterations for merging; $F$, fraction of $R_m$.
4:     Calculate the weight of each edge in the graph G based on Eq. (2).
5:     $DC \leftarrow \emptyset$
6:     repeat
7:         $S \leftarrow$ Sort edges based on non-decreasing order of their weights in graph $G$ (the edges with same values
8:              are listed in random order)
9:         $G'(V', E') \leftarrow G(V, E)$
10:        for $i = 1$ to $i \leq T$ do
11:            $e_r \leftarrow S[i]$
12:            if the degree of two end nodes of $e_r$ is not one then
13:                $E' \leftarrow E' - \{e_r\}$
14:            else
15:                $T \leftarrow T + 1$
16:            end if
17:        end for
18:        $C \leftarrow$ Each disconnected component in $G'$ (as primary community)
19:        Execute pairwise merging in $\frac{R_m}{F}$ iteration of primary communities
20:        Execute single neighbor merging in $R_m - \frac{R_m}{F}$ of primary communities
21:        $EF \leftarrow$ Calculate evolution function based on Eq. (7) for $C$ communities
22:        DC $\leftarrow$ EF and C
23:    until there not exists any non-repetitive partitioning on the graph
24:    Output: the community structure with the lowest value of evolution function ($EF$)
25: end procedure
```

6. Experiments and results

In this section, we analyze the performance of CDASS method based on implementation in MATLAB. In order to better evaluation, we compare our method with two previous methods of MST-M [5] and LPA-S [19] that are based on structural similarity and explained in Section 2. Therefore, we evaluate CDASS, MST-M and LPA-S methods using real network datasets based on three presented performance metrics.

In the following, we explain the properties of real and artificial network datasets that are used in our evaluation. In addition, we present evaluation measures that compare these methods based on them. Finally, we analyze the results obtained from our evaluation.

6.1. Dataset

In this section, we introduce several real and artificial networks that are used to evaluate CDASS method. In the following, we introduce these networks in details.

6.1.1. Real networks

We use three real networks [27] that their community structure is known.Therefore, these networks is suitable to evaluate community detection methods because according to these networks, we can compare our results with optimal communities. In the following, we explain several real networks that used in this article.

- Zachary's karate club network is a very popular network that used in several methods [5,19]. This network includes 34 clubs (nodes) and 78 edges that represent associations between members of the club at a university in the United States. In Zachary's karate club, all network clubs are included in two communities [28]. This database is notated as Zachary in the following tables.
- US College Football network shows the games' schedule among Division I of the US College Football League during the 2000 season. This network consists of 115 teams (nodes) and 613 edges. Each edge connects two teams that played together in this league. College football network is divided into 12 communities [29]. This database is notated as Football in the following tables.
- Political Books network is composed of American politics that compiled by Valdis Krebs during the 2004 presidential election. This network involves 105 books (nodes) and 441 edges. Each edge connects two nodes that their related books are purchased together. In this network, all nodes are divided into three communities [30]. This database is notated as Zachry in the following tables. This database is notated as Book in the following tables.

Table 1
The properties of used real network scenarios.

Real network name	Nodes	Edges
Zachary	34	78
Book	105	441
Adjnoun	112	425
Football	115	613
Jazz	198	2742
Email	1133	5451

We also extract several real networks from [31] that we show the properties of these networks in Table 1, including the number of network nodes and the number of edges.

6.1.2. *Artificial networks*

In order to better comparison, we also consider artificial networks to our evaluation. In this regard, we use Girvan–Newman (GN) artificial networks. In GN scenarios, each network is composed of 128 nodes and is divided into four communities including 32 nodes. In addition, the average degree of each network is 16 and approximately, all of nodes have the same degree. Moreover, the number of connections that each node has with other nodes located outside of its community is limited to parameter K_{out}. If K_{out} is lower than eight, each node has more connections with its community's nodes than the rest of network. Therefore, K_{out} is changed between one to eight in order to produce various networks.

6.2. *Evaluation measures*

In this section, we use three performance measures, including density, modularity and Normalized Mutual Information (NMI) metrics to evaluate the partitioning of the graph. These measures are introduced in the following.

- Density: Density function (D) is the sum of edges density within all communities in the graph. In each community, edges density is calculated by the number of internal edges respect to the total number of network edges. Eq. (10) shows the calculation of the density function.

$$D(\{V_i\}_{i=1}^{k}) = \sum_{i=1}^{k} \frac{|\{(v_x, v_y)|v_x, v_y \in V_i, (v_x, v_y) \in E\}|}{|E|} \tag{10}$$

  where $\{V_i\}_{i=1}^{k}$ represents k communities that are resulted from different methods and v_x and v_y are two nodes that belong to the same community V_i. In addition, $|\{(v_x, v_y)|v_x, v_y \in V_i, (v_x, v_y) \in E\}|$ is the number of edges that are included in community V_i and $|E|$ is the total number of edges in the network.
- Modularity: As introduced in Section 3.2, modularity measure is an optimization of quality function that evaluates quality of communities. In this regard, we use Newman–Girvan modularity to evaluate the proposed method. As mentioned, the larger value of modularity inducts the best quality of detected communities.
- Normalized Mutual Information: As mentioned, we evaluate our proposed method based on several real networks that their community structure is known. Therefore, NMI is a quality function that compares real communities in each network with resulted communities from proposed method. If community detection method exactly extracts real communities, the value of NMI measure will be one. In addition, if community detection method considers whole of the network graph as only one community, the value of NMI measure will be zero. In the following, we explain the calculation of NMI [3,24].
  To calculate NMI, a confusion matrix N is defined. In this matrix, the rows represent the real communities and columns is considered as resulted communities by the proposed method. Each element N_{ij} is the number of nodes in the real community i that are appeared in the resulted community j. In this regard, Eq. (11) shows the formula that is used for NMI calculation.

$$NMI(A, B) = \frac{-2\sum_{i=1}^{c_A}\sum_{j=1}^{c_B} N_{ij} log(\frac{N_{ij}n}{N_{i.}N_{.j}})}{\sum_{i=1}^{c_A} N_{i.} log(N_{i.}/n) + \sum_{j=1}^{c_B} N_{.j} log(N_{.j}/n)} \tag{11}$$

  where A and B are the sets of real communities and resulted communities, respectively. c_A is the number of real communities and c_B is the number of resulted communities from the proposed method. n is the number of network nodes. The sum over row i of matrix N_{ij} is defined $N_{i.}$ and the sum over column j is defined $N_{.j}$.

6.3. *Results*

As mentioned, we implement CDASS, LPA-S and MST-M methods in MATLAB based on several datasets that are introduced in Section 6.1. In this section, we analyze and compare obtained results using several evaluation measures that explained in Section 6.2.

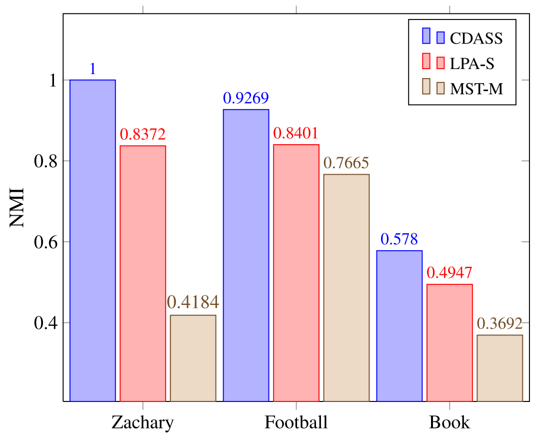

Fig. 1. Comparison among CDASS, LPA-S and MST-M algorithms, NMI in real networks.

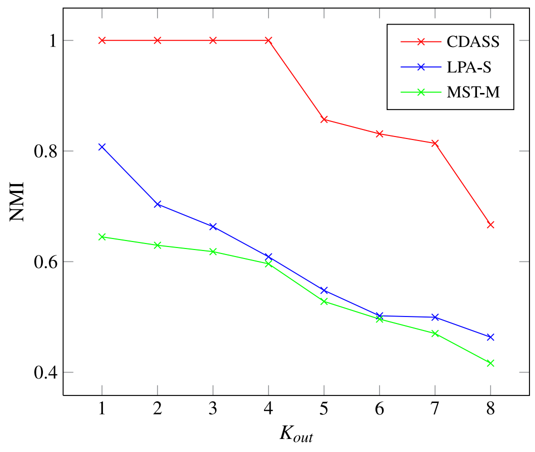

Fig. 2. Comparison among CDASS, LPA-S and MST-M algorithms, NMI in Girvan–Newman (GN) artificial networks.

Table 2
The modularity value in different real complex networks.

Network	CDASS	LPA-S	MST-M
Zachary	0.3715	0.3718	0.1986
Book	0.4936	0.4904	0.2012
Adjnoun	0.2532	0	0.1440
Football	0.6010	0.3855	0.2227
Jazz	0.4247	0.0169	0.0065
Email	0.5635	0.3645	0.2651

Table 3
The density value in different real complex networks.

Network	CDASS	LPA-S	MST-M
Zachary	0.8718	0.8717	0.3718
Book	0.9342	0.7800	0.2812
Adjnoun	0.5129	1	0.2541
Football	0.6917	0.4421	0.2577
Jazz	0.7414	0.0240	0.0073
Email	0.6850	0.3887	0.2976

6.3.1. Results in real networks

We use several datasets including US College Football, Zachary's karate club and Political Books networks to evaluate these methods using NMI measure. Fig. 1 shows the result of these methods. In this figure, CDASS method is more successful

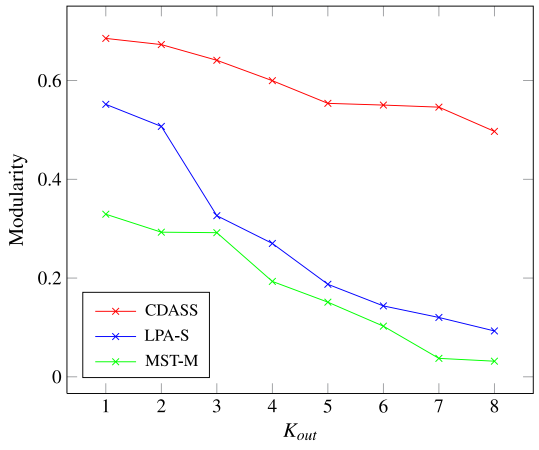

Fig. 3. The modularity value in Girvan–Newman (GN) artificial networks.

Table 4
The density value in Girvan–Newman (GN) artificial networks.

K_{out}	CDASS	LPA-S	MST-M
1	0.9353	0.7241	0.1628
2	0.8652	0.8652	0.1080
3	0.8131	0.1789	0.0646
4	0.7720	0.2280	0.0812
5	0.8816	0.1373	0.0716
6	0.9756	0.1696	0.0820
7	0.8338	0.1503	0.0692
8	0.9768	0.110	0.0550

than LPA-S and MST-M methods in community detection, because the NMI value of this method is more than others in all the networks.

To better evaluation, we also execute these three methods on all of network datasets that introduced in Section 6.1 from the point of view of modularity and density measures. Table 2 shows the modularity measure of these methods. Results show that the modularity value of CDASS method is more than other methods. Therefore, CDASS detects communities similar to optimal version. Table 3 displays the density measure of CDASS, LPA-S and MST-M methods. This table shows that the density of CDASS method is more than other methods. Then, CDASS method is better than other methods.

6.3.2. *Results in artificial networks*

We also evaluate CDASS method using GN network scenarios. The results of these evaluations are shown in Figs. 2, 3 and Table 4. The Fig. 2 shows the amount of NMI measure. In this figure, the NMI value in case of K_{out} between one to four is 1 whereas LPA-S and MST-M do not achieve good results. In addition, the NMI value in the case of $K_{out} = 8$ is 0.6667, therefore CDASS method has high accuracy compared with other works. In the figure, modularity measure is shown. So that, the modularity value of CDASS method is good and better than other methods. Finally, Table 4 shows the density of Girvan–Newman (GN) artificial networks. CDASS method has the better density than LPA-S and MST-M methods. Moreover, in all GN networks, CDASS has density more than 0.7. Finally, these results represent that our proposed method has been succeeded and detect communities in the best situation.

6.4. *Discussion*

The MST-M, LPA-S and CDASS algorithms use different similarity measure to detect community. The MST-M algorithm remove edges step by step and convert network to a tree. Afterwards, some other edges are removed from resulted tree to generate several primary communities. In this regard, there exist some primary communities composed of only one node, while CDASS algorithm tries to remove inter community edges and each primary community includes at least two nodes. In LPA-S algorithm, the similarity is computed for each pair of nodes. Therefore, this method has to spend significant time to compute similarity and is not applicable in the large networks.

In order to better evaluation, we compare these algorithms in case of time complexity. As mentioned in Section 5, the time complexity of CDASS is $O(m \log m + R(m + n + m \log n + m))$. R is very low in execution mode, the time complexity of CDASS can be approximately shown by $O(m \log m)$. In addition, the time complexity of LPA-S and MST-M are reported by $O(n^2)$ [19] and $O(2m + n(\log n + (\frac{n-1}{2})m + |C_r|))$ [5], respectively. Where $|C_r|$ is the number of community with one node. Therefore, to better comparison, the time complexity of MST-M algorithm can be shown by $O(n^2m)$. Consequently, the time complexity of CDASS, LPA-S and MST-M algorithms is $O(m \log m)$, $O(n^2)$ and $O(n^2m)$, respectively. According to real network, the average degree of each node is very lower than n therefore we can approximate m by $O(n)$. Therefore, the time complexity of CDASS, LPA-S and MST-M algorithms can be approximately modeled by $O(n \log n)$, $O(n^2)$ and $O(n^3)$. Therefore, our proposed algorithm has better complexity compared to others in the real networks.

7. Conclusions

We proposed CDASS method, a novel mechanism to detect communities based on structural similarity, that is executed in two consecutive phases. This method converts the network graph into several disconnected components by random removing some low similarity edges. Afterwards, primary communities are merged in order to identify the final community structure close to real communities. Finally, we use an our identified evaluation function to select the best communities between overall random generated partitions. Remarkable points of this method are randomly removing edges and generating of primary communities with at least two nodes. In randomly removing edges, some edges with low similarity are selected to remove. In this regard, resulting edges that have same similarity are selected randomly to remove. In generating of primary communities with at least two nodes, primary communities are produced at least two nodes. In this regard, it prevents the creating of single-node communities and the time complexity of merging improves. Results obtained from several real and artificial network scenarios show that this method is successful in large and small datasets. Therefore, results obtained from evaluation measures such as NMI showed that CDASS method has high accuracy. In addition, The time complexity of CDASS method is low and this method is applicable in the large complex networks. Some directions that can be investigated are how to extend CDASS method to consider weighted complex networks. Moreover, this method can be extended to networks that are based on attribute similarity.

References

[1] S. Fortunato, Community detection in graphs, Phys. Rep. 486 (3) (2010) 75–174.
[2] J. Leskovec, K.J. Lang, M. Mahoney, Empirical comparison of algorithms for network community detection, in: Proceedings of the 19th International Conference on World Wide Web, ACM, 2010, pp. 631–640.
[3] P. De Meo, E. Ferrara, G. Fiumara, A. Provetti, Mixing local and global information for community detection in large networks, J. Comput. System Sci. 80 (1) (2014) 72–87.
[4] M. Arab, M. Afsharchi, Community detection in social networks using hybrid merging of sub-communities, J. Netw. Comput. Appl. 40 (2014) 73–84.
[5] B. Saoud, A. Moussaoui, Community detection in networks based on minimum spanning tree and modularity, Physica A 460 (2016) 230–234.
[6] M.P. Boobalan, D. Lopez, X. Gao, Graph clustering using k-neighbourhood attribute structural similarity, Appl. Soft Comput. 47 (2016) 216–223.
[7] M. Ciglan, M. Laclavík, K. Nørvåg, On community detection in real-world networks and the importance of degree assortativity, in: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2013, pp. 1007–1015.
[8] J. Yang, J. McAuley, J. Leskovec, Community detection in networks with node attributes, in: Proceedings of the 13th IEEE International Conference on Data Mining, ICDM2013, IEEE, 2013, pp. 1151–1156.
[9] M.E. Newman, M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E 69 (2) (2004) 026113.
[10] V.D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech. Theory Exp. 2008 (10) (2008) P10008.
[11] U. Brandes, D. Delling, M. Gaertler, R. Görke, M. Hoefer, Z. Nikoloski, D. Wagner, Maximizing modularity is hard, 2006. arXiv preprint physics/0608255.
[12] Y. Pan, D.-H. Li, J.-G. Liu, J.-Z. Liang, Detecting community structure in complex networks via node similarity, Physica A 389 (14) (2010) 2849–2857.
[13] T. Wang, L. Yin, X. Wang, A community detection method based on local similarity and degree clustering information, Physica A 490 (2018) 1344–1354.
[14] C. Piccardi, Finding and testing network communities by lumped markov chains, PLoS One 6 (11) (2011) e27028.
[15] D. Jin, B. Yang, C. Baquero, D. Liu, D. He, J. Liu, A Markov random walk under constraint for discovering overlapping communities in complex networks, J. Stat. Mech. Theory Exp. 2011 (05) (2011) P05031.
[16] Y. Xin, Z.-Q. Xie, J. Yang, An adaptive random walk sampling method on dynamic community detection, Expert Syst. Appl. 58 (2016) 10–19.
[17] M.J. Barber, J.W. Clark, Detecting network communities by propagating labels under constraints, Phys. Rev. E 80 (2) (2009) 026129.
[18] L. Šubelj, M. Bajec, Ubiquitousness of link-density and link-pattern communities in real-world networks, Eur. Phys. J. B 85 (1) (2012) 1–11.
[19] W. Li, C. Huang, M. Wang, X. Chen, Stepping community detection algorithm based on label propagation and similarity, Physica A 472 (2017) 145–155.
[20] X.-K. Zhang, J. Ren, C. Song, J. Jia, Q. Zhang, Label propagation algorithm for community detection based on node importance and label influence, Phys. Lett. A 381 (33) (2017) 2691–2698.
[21] J. Reichardt, S. Bornholdt, Statistical mechanics of community detection, Phys. Rev. E 74 (1) (2006) 016110.
[22] B. Karrer, M.E. Newman, Stochastic blockmodels and community structure in networks, Phys. Rev. E 83 (1) (2011) 016107.
[23] J.P. Bagrow, Evaluating local community methods in networks, J. Stat. Mech. Theory Exp. 2008 (05) (2008) P05001.
[24] L. Danon, A. Diaz-Guilera, J. Duch, A. Arenas, Comparing community structure identification, J. Stat. Mech. Theory Exp. 2005 (09) (2005) P09008.
[25] Z. Liu, P. Li, Y. Zheng, M. Sun, Community Detection by Affinity Propagation, Tech. Rep., Technical Report, 2008.
[26] T.H. Cormen, C. Stein, R.L. Rivest, C.E. Leiserson, Introduction to Algorithms, second ed., McGraw-Hill Higher Education, 2001.
[27] M.N. personal.umich.edu/~mejn/netdata/ (online).
[28] W.W. Zachary, An information flow model for conflict and fission in small groups, J. Anthropol. Res. 33 (4) (1977) 452–473.
[29] M. Girvan, M.E. Newman, Community structure in social and biological networks, Proc. Natl. Acad. Sci. 99 (12) (2002) 7821–7826.
[30] F. Souam, A. Aïtelhadj, R. Baba-Ali, Dual modularity optimization for detecting overlapping communities in bipartite networks, Knowl. Inf. Syst. 40 (2) (2014) 455–488.
[31] http://www.cc.gatech.edu/dimacs10/archive/clustering.shtml (online).