Applied Soft Computing Journal 89 (2020) 106096

Contents lists available at ScienceDirect

Applied Soft Computing Journal

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

InOvIn: A fuzzy-rough approach for detecting overlapping communities with intrinsic structures in evolving networks

Keshab Nath [a], Swarup Roy [b,*], Sukumar Nandi [c]

[a] Department of Information Technology, North Eastern Hill University, Shillong, 793022, India
[b] Department of Computer Applications, Sikkim University, Sikkim, 737102, India
[c] Department of Computer Science and Engineering, Indian Institute of Technology, Guwahati, 781039, India

ARTICLE INFO

Article history:
Received 26 November 2018
Received in revised form 29 November 2019
Accepted 13 January 2020
Available online 30 January 2020

Keywords:
Incremental community
Embedded cluster
Density variation
Community within community
Overlapping community
Rough-fuzzy set
Social graphs

ABSTRACT

Real-world networks, such as biological, biomedical and social networks, often contain overlapping and intrinsic communities. More significantly, such networks are growing or evolving over time, which leads to a continuous alteration of community structures. Detecting overlapping community together with intrinsic structures in evolving scenarios is one of the challenging tasks. Prior researches are limited in handling all the events together while designing a community detector.

We propose an integrated solution, InOvIn (**In**trinsic **Ov**erlapping Community Detection in **In**cremental Networks), for detecting overlapping, non-overlapping and intrinsic communities in evolving networks. Herein, we have explored a rough-fuzzy clustering approach for overlapping community detection. Fuzzy membership helps in soft decision making for deciding membership of a node towards a target community. While rough boundary of the communities decides the shared membership of a node in multiple communities. The node degree density variation measure is used to discover the existence of intrinsic community within a community.

We assess the performance of InOvIn in light of twelve (12) popular real-world social networks. It may be noted that available real-world networks are lacking in labeled overlapping and intrinsic communities. Hence, we synthetically generate six (06) networks with both overlapping and intrinsic communities. We demonstrate the superiority of InOvIn over contemporary community detection methods using ten (10) different statistical assessment parameters. Interestingly, for the first time, our method detects intrinsic communities in *PolBooks* and *Word Adjacencies* networks.

1. Introduction

Rapid changes are occurring among the entities in any complex system due to the change in subjects of interest and relationship. As a outcome, the networks based on those entities are also evolving. Genetic networks, brain networks, protein interaction networks, and social networks are some of the striking examples of evolving networks. Like other networks, social network is also mapped as a graph, where objects (users) are represented as a set of vertices and the interaction between objects are represented as edges. Nodes with similar characteristics or functions usually brought together to single unit (sub-graph), termed as *network communities* or *modules*. Nodes in a community usually exhibit high intra-community connectivity and less inter-community connectivity.

In an evolving network, a community may pass through several phases of transformation over time. Due to frequent changes in the associations or choice of interests, an actor or a user (client) may develop an affinity towards different interest groups. It may leads to shared participation of a user in more than one group, simultaneously. Communities that share common members across different communities are frequently termed as *overlapping communities*. The spontaneous growth of the network and relationships among the nodes may create a trend of forming more compact sub-community within a community, which we termed it as *intrinsic communities* (also termed as embedded communities). The intrinsic community may exhibit varying intra-community connectivity (high or low) differing from the parent community it belongs to. Formation of intrinsic communities is more common in genetic networks [1], protein–protein interactions [2–4] and metabolic networks [5–7]. The being of such communities is also reported even in social networks [8,9].

For the last few decades, efforts are on for effective community detection in both static and dynamic networks. Most of the study concentrates on finding disjoint (or non-overlapping) communities with an assumption that a node may exclusively be the member of one community only and networks are not

* Corresponding author.
E-mail addresses: keshabnath@nehu.ac.in (K. Nath), sroy01@cus.ac.in (S. Roy), sukumar@iitg.ernet.in (S. Nandi).

https://doi.org/10.1016/j.asoc.2020.106096
1568-4946/

altered over time. Very few attempts have been made in detecting overlapping [10,11] and intrinsic communities [12,13] in dynamic or evolving networks [14,15]. However, they do not direct all three issues simultaneously. There is a need for a community detection method that can offer an integrated solution for the detection of overlapping as well as intrinsic communities in evolving networks. Traditional approaches are not effective in overlapping community detection when relationships are imprecise. Soft computing paradigm is an effective option to handle imprecise and uncertain scenarios. In this study, we address the issue of detecting overlapping communities in evolving networks with the help of fuzzy-rough clustering approach. The concept of community density variation has been applied to detect any possible existence of intrinsic communities. The contributions of our work are listed below.

- We propose an integrated overlapping and intrinsic community detection method in evolving networks. A fuzzy membership function is proposed to handle the imprecise affinity of a node towards a community which varies over time.
- Concept of rough boundaries, derived from the rough set is applied to identify the non-overlapping and overlapping nodes within multiple communities.
- Node degree density variations within a community are considered to detect the formation of intrinsic communities over time.
- Due to unavailability of appropriate benchmark networks with overlapping and intrinsic communities, we synthetically generate networks with both overlapping and intrinsic structures.
- The superiority of `InOvIn` over contemporary community detection methods is established using ten (10) different statistical assessment parameters for twelve (12) real-world benchmark social networks and six (6) synthetically generated networks.
- Proposed method able to detect the existence of intrinsic communities in *PolBooks* and *Word Adjacencies* networks for the first time.

The rest of the paper is organized as follow. In Sections 2 and 3, we formally introduce the background of the problem and present different prior attempts on handling overlapping and intrinsic communities in evolving networks. In Section 4, proposed method has been discussed. Performance evaluation of the proposed method is reported in Section 5. Eventually, in Section 6 the paper is summarized with concluding remarks.

2. Background

Graph-theoretic formalism is the most common way of representing a social network. A network community is a cluster [16] or cohesive subgraph with significant intra-community connectivity among the nodes of a community than nodes of other communities in the network. Formally, it may be defined as follows.

Definition 2.1 (*Network Community*)**.** Assume network $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ represent a set of nodes and edges respectively. A community $C_i = \{V_i, E_i\}$ ($V_i \subseteq \mathcal{V}$ and $E_i \subseteq \mathcal{E}$) can be defined as a sub-graph of $\mathcal{G}$, where membership score (based on some parameter like relation, interest, etc.) of each $v_i \in V_i$ is higher with C_i compared to any other community C_j.

The above definition of the network community relies on classical definition of disjoint clusters, where it is assumed that the members of a community cannot be a member of any other communities simultaneously i.e. $V_i \cap V_j = \phi$ ($V_i \in C_i$ and $V_j \in C_j$). However, it is very unrealistic to consider that communities are solely exclusive. In social graphs, a user may participate in more than one interest group or contexts, showing overlapping memberships. Formally, we define overlapping communities as follows.

Definition 2.2 (*Overlapping Community*)**.** Two communities $C_i = \{V_i, E_i\}$ and $C_j = \{V_j, E_j\}$ are overlapping, if $V_i \cap V_j \neq \phi$ whereas $E_i \cap E_j = \phi$.

Detecting overlapping communities in any network is a challenging task. This is because overlapping nodes possess imprecise membership among multiple communities. It is even more difficult when the network is dynamic where membership of a node across different communities varies over time. Traditional community detection methods are limited to handle static networks and do not consider the dynamic scenarios where users frequently change their relationship and communities over time. It is relatively easy to design a static community detector where nodes are assumed to belong to only one community. However, such methods are inadequate in the case of dynamic networks, where networks are continually evolving. Due to the rapid and continuous evolution of the network, nodes might shift from one community to another, frequently. As a consequence, there may be the formation of a new community by splitting existing community or merging of two communities into one. The incremental clustering approach may be the possible way to handle such situations by processing each of the incoming nodes in real-time to decide their membership. An incremental community clustering can be defined as follows.

Definition 2.3 (*Incremental Community Clustering*)**.** Given an incremental network, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the incremental community clustering is a mapping function $f : \mathcal{G} \rightarrow \{1, \ldots, k\}$ such that each incoming v_i is assigned to a community $\mathcal{C}_i(t)$ at time t. Community $\mathcal{C}_i(t)$ contains those members that are mapped to it at time t. At time $t + 1$, any community $\mathcal{C}_i(t + 1)$ may contain object $v_p \in \mathcal{C}_j(t)$ and $\mathcal{C}_j(t + 1) = \mathcal{C}_j(t) - v_p$ ($\forall j = 1, \ldots, k$).

In the above definition, the number of communities, k, is dynamic and may decrease or increase depending on the incoming nodes and their association with other existing nodes in different communities.

The effectiveness of an incremental community detection method largely depends on the order of the incoming nodes. It causes movement of nodes from one to another community with time. Importantly, the outcomes of any incremental method should not differ much from its static version. An illustration of a probable situation of node movements from one community to another is explained in the supplementary material (*ref* Sec. 2, Fig. 1).

With the continuous growth of the network, a different overlapping community may form within a community, referred to as an embedded or *intrinsic community*. Intrinsic communities are the compact community embedded in the parent community with a significant connectedness density variation, concerning their neighborhood connectivity within the group.

Definition 2.4 (*Connectedness Density*)**.** For a community, $C_i = \{V_i, E_i\}$, the connectedness density of the community, $\rho(C_i)$, is the ratio between the total number edges and the total number of nodes within the community C_i. It can be defined as follows.

$$\rho(C_i) = \frac{\sum_{p=1}^{|V_i|} \sum_{q=1}^{|V_i|} \lambda(v_p, v_q)}{|V_i|}. \tag{1}$$

where,

$$\lambda(v_p, v_q) = \begin{cases} 1, & \text{if } (v_p, v_q) \text{ is connected}, \forall v_p, v_q \in V_i \\ 0, & \text{Otherwise} \end{cases}$$

Based on the above connectedness density, we may define an intrinsic community as follows.

Definition 2.5 (*Intrinsic Community*)**.** A community C_i is intrinsic or embedded inside C_j, if $C_i \subset C_j$ and connectedness density of C_i is significantly higher than C_j with respect to a certain threshold, θ, i.e., $|\rho(C_i) - \rho(C_j)| > \theta$.

Examples of intrinsic, overlapping and non-overlapping community structures are shown in Fig. 1.

Next, we discuss few state-of-the-art methods for disjoint and overlapping community detection in both static and dynamic networks. Moreover, we also discuss some of the recent intrinsic community detection algorithms.

3. Prior research

Most of the prior work given attention more on detecting overlapping and non-overlapping communities. Some of them even handle dynamic network scenarios. We discuss here different community detection methods based on their working principles and the type of communities they detect.

3.1. Non-overlapping community detectors

Infomap [17] employs random walks to maximize the amount of information flow on the decomposed sub-networks for community detection. Yu-Hsiang Fu et al. [18] introduce a rule-based arc-merging strategies for identifying community structures. The hierarchical arc-merging (HAM) architecture consists of similarity measurement and modularity optimization phases and rule-based strategies for community detection. DyPerm [14] is a dynamic community detection method which incrementally modifies the existing communities by optimizing a community scoring metric, called permanence. iDBLINK [19] is an incremental density-based community detection algorithm in dynamic networks. Based on the changes in the connections between nodes over time, it immediately updates the corresponding community structures. Inspired from Fuzzy Granular Social Networks (FGSN) [20] and Louvain algorithm [21], Nicole et al. [22] proposed a method for detecting communities in a social network. However, the above methods do not build to handle overlapping communities. Moreover, except for a few, most of the approaches consider the network as a static.

3.2. Overlapping community detectors

Overlapping Cluster Generator (OCG) [23] creates a hierarchy of overlapping clusters according to an extension of Newman's modularity function. Preferential Learning and Label Propagation Algorithm, called PLPA [24] is proposed to detect overlapping communities based on learning behavior and information interaction in social networks. COPRA [25] optimizes and expands seed community iteratively based on the clustering coefficient of each node by taking the average clustering coefficient of neighboring nodes. LPANNI [26], is a label propagation-based method to detect overlapping communities. It quantifies the influence of neighbor nodes by adopting node importance and affinity between node pairs to improve the label update strategy. SLPAD [11] follows label propagation model like SLPA [15] for detecting overlapping communities in dynamic networks. Shuai Ding et al. proposed a new trust model based overlapping community detection algorithm called TLCDA [27]. They improve the conventional trust computation with inter-node relation strength and similarity in social networks, then perform community detection through coarse-grained K-Mediods clustering. AFOCS [28] is an adaptive framework for the detection of overlapping communities as well as tracking the evolution of overlapping communities in incremental networks. TILES [29] extracts overlapping communities in dynamic social networks using peripheral membership and core membership and re-evaluate node's community membership for each new interaction. OSLOM [30] uses local optimization of a fitness function to detect overlapping communities in incremental networks. An Extended Adaptive Density Peaks clustering (EADP) [31] is proposed for overlapping community detection based on a novel distance function. EADP adaptively choose cluster centers by employing a linear fitting based strategy.

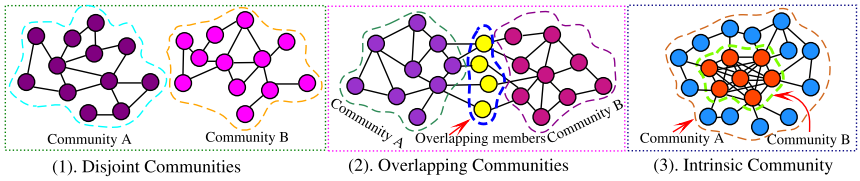
A fuzzy-rough non-incremental community detection method is proposed [20] for detecting overlapping communities based on fuzzy granular theory. Hao Wu et al. [32] present a novel method based on the rough-fuzzy clustering to detect overlapping and non-overlapping protein complexes in PPI networks. All the methods discussed above are primarily focused on detecting overlapping communities in either static or dynamic networks. Yet, they have not reported the presence of intrinsic communities in evolving networks.

3.3. Intrinsic and overlapping community detectors

EAGLE [13] able to uncover both overlapping and intrinsic communities by finding the maximum cliques followed by agglomerative clustering. Ahn et al. [12] proposed a model to discover link communities to reveal the multiscale complexity present in a network. iLCD [10] is introduced for finding overlapping and intrinsic communities in incremental networks. Though iLCD claimed to handle intrinsic communities, however, it mainly focuses on detecting overlapping communities in dynamic networks. Recently, InDEN [33] has been proposed to detect intrinsic communities in growing networks. However, they do not consider overlapping structures. A summary of the various methods discussed above is reported in Table 1. For more extended theoretical comparison of the state-of-the-art community detectors, one can refer to Sec. 3, Table. 1 in the supplementary material.

Discussion: Existing methods address the issue of overlapping and intrinsic community detection in evolving networks independently. Some of them focus only on overlapping communities. A few of them adequate to detect both disjoint and overlapping communities in dynamic networks. However, to the best of our knowledge, no single work addresses the issue of handling disjoint, overlapping as well as embedded communities in evolving networks in an integrated way.

Traditional clustering methods are *crisp* in nature and incompatible for finding overlapping network communities. Soft computing techniques, more precisely fuzzy [36] and rough clustering [37] approaches are a better alternative to detect overlapping communities in the presence of uncertainty. Rough-fuzzy hybrid models are even more effective in handling similar situations. Other than social graph mining [20], rough-fuzzy models are equally popular in biological community detection [32]. However, available rough-fuzzy models are not suitable for growing or incremental networks, which is a very significant factor from a social network point of view. Besides, as stated above, due to the varying connectivity within a community may contribute to the formation of intrinsic communities. An integrated solution for detecting disjoint, overlapping and intrinsic communities in evolving networks is our prime objective that has been handled next.



(1). Disjoint Communities (2). Overlapping Communities (3). Intrinsic Community

Fig. 1. Representation of Disjoint, Overlapping and Intrinsic Communities. **(1)** Denotes the presence of two disjoint communities. Yellow color nodes in **(2)** are the members showing common interests in both the communities. In **(3)** Community B is embedded inside the community A.

Table 1
A comparative analysis of existing community detection algorithms.

Algorithms	Overlapping	Intrinsic	Incremental	Availability	Technique used
SLPAD (2014) [11]	✓		✓		Label propagation
Kundu et. al. (2015) [20]	✓				Fuzzy-rough set
EAGLE (2009) [13]	✓	✓		GitHub[a]	Maximal cliques
iLCD (2010) [10]	✓		✓		Longitudinal analysis
CONGA (2007) [34]	✓				Betweenness centrality measure
OSLOM (2011) [30]	✓		✓	Oslom[b]	Fitness function
COPRA (2010) [25]	✓			GitHub[c]	Label propagation
OCG (2012) [23]	✓			linkcomm[d]	Modularity
TILES (2017) [29]	✓		✓	TILES[e]	Community memberships
DyPerm (2018) [14]			✓	DyPerm[f]	Community Scoring Metric
HAM (2017) [18]				HAM[g]	Rule-Based Arc-Merging
LPANNI (2018) [26]	✓			–	Label Propagation
InDEN (2019) [33]		✓	✓	–	Community memberships
Infomap (2008) [17]				MapEquation [h]	Map Equation
iDBLINK (2016) [19]			✓	Dimensions[i]	Density-based
LabelRankT (2013) [35]			✓	GitHub[j]	Label propagation
AFOCS (2011) [28]	✓		✓	Optnetsci[k]	Adaptive framework
Yong-Yeol Ahn *et al.* (2010) [12]	✓	✓		–	Maximal partition density

[a] https://github.com/RapidsAtHKUST/CommunityDetectionCodes.
[b] http://www.oslom.org/.
[c] https://github.com/RapidsAtHKUST/CommunityDetectionCodes/tree/master/Algorithms/2010-CONGA/conga_src/src/main/java/copra/algorithm.
[d] https://rdrr.io/cran/linkcomm/.
[e] https://github.com/GiulioRossetti/TILES.
[f] https://github.com/ayush14029/Dyperm-Code.
[g] https://github.com/yuhsiangfu/Hierarchical-Arc-Merging.
[h] http://www.mapequation.org/code.html.
[i] https://app.dimensions.ai/details/publication/pub.1000284141.
[j] https://github.com/RapidsAtHKUST/CommunityDetectionCodes/blob/master/Survey/Overlapping-Community-Detection-Codes.md.
[k] http://optnetsci.cise.ufl.edu/src/.

4. An integrated model for effective community detection

We develop a community detection model, considering the evolving scenario of a network, using the well-known fuzzy-rough clustering concept. Rough set [37] is good in effective decision making in a situation of uncertainty. It utilizes the concept of lower (strong) and upper (weak) approximation in an approximation space. Lower approximation probabilistic helps to discover the community shape. Objects that belong to a lower approximation of a community are indiscernible and exclusively belong to a particular community. Any node may belong to a lower boundary region of a community only when membership in the community is substantial. On the other hand, it may belong to upper approximation, when it shares relatively equal belongings strength with more than one community. We use a fuzzy membership function to define the association of a node to the community.

Membership function (μ) measures the relationship strength of a node towards a community. We consider a shared node degree with existing communities at time t. Sharing node degree of a node (towards a community) is the ratio between the total degree of the node and the number of neighbors of the node within the target community. It means the belongings of a node v_k towards a community C_i is decided by the neighbors of that node. Therefore, the membership of a node is decided by the connectivity strength of a node towards a community. We consider two aspects while calculating the membership of a node. Firstly, the total number of adjacent nodes (degree) of a node, say v_k is in C_i in comparison to the total size of the community, C_i. Secondly, out of the total degree of v_k ($\delta(v_k)$) how many are in C_i. Formally, it can be defined as follows.

$$\mu_{C_i}(v_k) = \frac{\sum_{i=1}^{|C_i|} \lambda\,(v_k, v_i)}{|C_i|} \times \frac{\sum_{i=1}^{|C_i|} \lambda\,(v_k, v_i)}{\delta(v_k)} \tag{2}$$

In case of growing network, the value of both $\sum_{i=1}^{|C_i|} \lambda\,(v_k, v_i)$ and $\delta(v_k)$ are not fixed and may alter due to variation in connectivity.

To illustrate the fuzzy behavior of the above equation, we take the help of Fig. 2. Let us consider a node $v_i \in V$ having degree 6 and it belongs to a community C_i, with high membership at time t. At time $t+1$ a new direct neighbor of v_i, say v_j appears in another community C_j. It leads to a decrease in membership strength of v_i to C_i and gain membership strength towards community C_j. When we keep on decreasing connectivity strength of node v_i, accordingly membership score gradually decreases towards C_i and increases with C_j. At the time $t+1$, if the difference

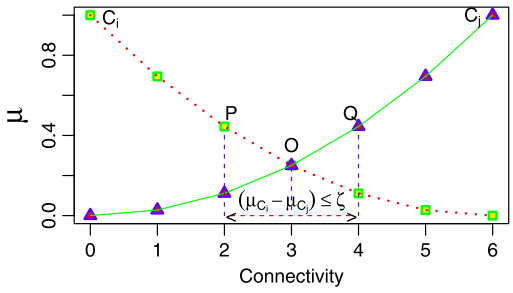


Fig. 2. Behavior of fuzzy membership function with varying connectivity strength. The graph illustrates the varying membership score of any particular node, v_i, for two different communities, C_i (dotted curve) and C_j (solid curve), at different time points. P, O, Q indicates state of v_i when membership scores are uncertain with respect to ζ, and having almost equal membership in both the communities. Score at O is the most uncertain with equal neighbors (3 in each) in both C_i and C_j.

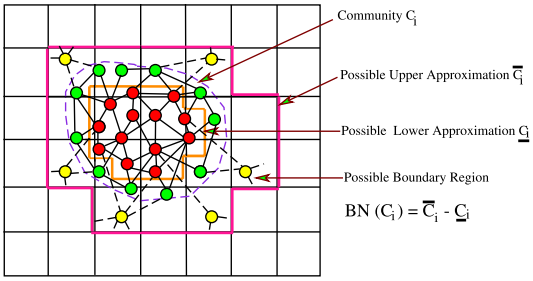


Fig. 3. The relationship among community C_i and its possible lower approximation, upper approximation and boundary region for the equivalence relation μ.

of the membership scores of v_i towards C_i and C_j is within ζ, then the node is considered to be an overlapping node with almost equal connectivity in both the communities. If we further decrease the connectivity of v_i with C_i, then node v_i shift to community C_j as because v_i gains more membership strength towards C_j than C_i.

Based on the above fuzzy membership, we develop the fuzzy-rough clustering model. A node v_i belongs to a lower boundary of a community only when membership of v_i concerning a community C_i ($\mu_{C_i}(v_i)$) is greater than a threshold (η), otherwise, it belongs to the upper boundary of C_i. When the difference of membership of node v_i to the community C_i and C_j is insignificant i.e. less than a threshold value (ζ), indicates v_i having approximately same belongings towards both the communities. We consider all such nodes present in the upper boundary as overlapping nodes of both C_i and C_j. The lower ($\underline{C_i}$) and upper boundary ($\overline{C_i}$) for any community C_i are two sets that can be represented as follows.

$$\underline{C_i} = \{v_i | v_i \in V, \mu_{C_i}(v_i) \geq \eta \text{ and } \forall_{j=1,\ldots,n},\ \mu_{C_j}(v_i) \leq \eta\} \tag{3}$$

$$\overline{C_i} = \{v_i | v_i \in V,\ |\mu_{C_i}(v_i) - \mu_{C_j}(v_i)| \leq \zeta \text{ and } \mu_{C_i}(v_i) \leq \eta\} \tag{4}$$

A community C_i is *crisp*(or non overlapping) only when the boundary region is empty. The boundary region of C_i can be defined as $BN_{C_i} = \overline{C_i} - \underline{C_i}$. If a boundary region of $C_i \neq \phi$, then C_i is considered as *rough* (or overlapping) community. We illustrate the approximation of a community in Fig. 3.

Next, we discuss our proposed intrinsic community detection method using the concept of node degree density variation.

4.1. Detecting intrinsic communities

Continuous insertion of edges in a community may create a new dense region within the community forming intrinsic or embedded community structures (an illustration is rendered in the Supplementary material(Sec. 4.2, Fig. 2)). Real-world networks usually follow the properties of *scale-free network* [38] and node connectivity within the network exhibits *power-law distributions*. Due to the *preferential attachment* of scale-free networks, nodes tend to associate more with the hub or core nodes and connectivity becomes sparse as it moves towards the boundary. Therefore, if there is a sufficient number of nodes which are highly connected within a community, then there is a possibility of connectedness density variation in that community. To detect such communities we use the degree density variations within a community. Instead of using Eq. (1), we consider a different concept to detect density and its variation within a community. The density variation is identified with the help of degree distribution curves at different time intervals.

We represent each community of size m (nodes) with a *cluster feature vector* $CF_{C_i} = \{\delta(v_1), \delta(v_2), \ldots, \delta(v_m)\}$, where, $\delta(v_j)$ is the degree of a node $v_j \in C_i$. For every insertion of a new node in C_i, CF_{C_i} is updated accordingly. Affected $\delta(v_j)$ ($\forall j \in CF_{C_i}$) is incremented if node v_j has a new incoming edge (v_j, v_k). Accordingly, the CF_{C_i} is updated to accommodate $\delta(v_k)$ for the new node v_k. We try to detect possible density variation in a community in time $t+1$, based on a significant change (if any) happens within the area under the corresponding connectedness density curve. Given feature vector $CF_{C_i}^{t+1}$ at time $(t+1)$, density variation occurs if the following inequality holds.

$$\overline{CF}_{C_i}^{t+1} > max(CF_{C_i}^{t+1}) - \xi \times \sigma(CF_{C_i}^{t}) \tag{5}$$

where ξ is the curve spread factor, $\overline{CF}_{C_i}$ and σ is mean and standard deviation of the feature vector of C_i respectively and can be calculated as follows.

$$\overline{CF}_{C_i} = \frac{1}{|C_i|} \sum_{i=1}^{|C_i|} \delta(v_i) \tag{6}$$

$$\sigma(CF_{C_i}) = \sqrt{\frac{1}{|C_i|} \sum_{i=1}^{|C_i|} (\delta(v_i) - \overline{CF}_{C_i})^2} \tag{7}$$

To illustrate the fact we consider degree distribution curve (Fig. 4) of a community at time t and $t+1$ with sparse degree distribution and density variation in time $t+1$ after the addition of new nodes and edges respectively. We try to detect variation in density with the help of changes in the curve spread. This can be accomplished by considering changes in the mean and standard deviation of the curve. For example, let us assume a degree distribution for a community at time t and $t+1$ as {1, 1, 3, 8, 9, 8, 7, 3, 2, 2} and {2, 3, 4, 8, 10, 11, 12, 13, 15, 15, 14, 13, 12, 10, 7, 6, 5, 4, 3, 2} respectively. The mean and standard deviation of the curve at t is $\overline{CF}_{C_i}^{t} = 4.4$ and $\sigma^t = 3.20$ respectively. At time $t+1$, $\overline{CF}_{C_i}^{t+1}$ is 8.45 and σ^{t+1} is 4.57. Since, δ_{max}^{t+1} (maximum degree) for the second distribution curve is 15 and $\xi = 2.1$, hence the inequality as in Eq. (5) holds and it confirms the occurrence of density variation within the community at $t+1$. A preliminary version of the work can be found in [39].

In addition to the above density variation, two more conditions may arise on continuous arrival of nodes or edges in a community.

- Due to successive insertion of new nodes inside a community cause an inter-community migration of the nodes. As a result, two communities may merge to form a single community or an existing community may split into two different communities.
- Nodes may acquire an equal membership score in multiple communities to form overlapping communities.

`InOvIn` handle such situations too, which we discuss next.

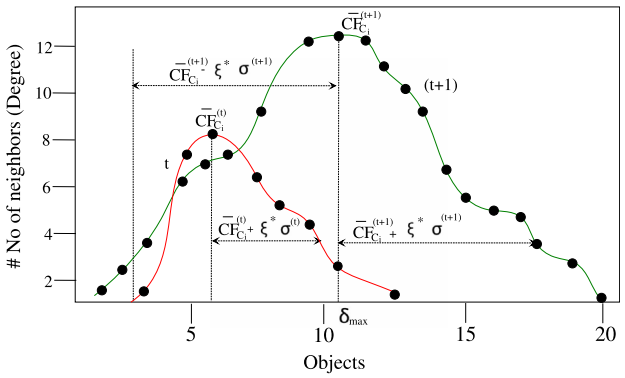


Fig. 4. Illustration of density variation with respect to density curve for 20 different nodes at t (red) and $t+1$ (green) time.

4.2. InOvIn : The algorithm

InOvIn initializes community set C with a community C_1 containing very first two nodes v_1 and v_2 of the first edge $e(v_1, v_2)$. In successive addition of new edges into the network over the period of time, the networks alter accordingly. For any incoming edge $e(v_i, v_j)$ in the network $\mathcal{G}(t+1)$, at time $t+1$, one of the following situations may arise.

1. A new edge is introduced into the network, $\mathcal{G}(t+1)$, where both v_i and v_j are existing in $\mathcal{G}(t)$. Node v_i and v_j may belong to the same community or may not.
2. A new edge $e(v_i, v_j)$ is added to $\mathcal{G}(t+1)$, where both v_i, $v_j \notin \mathcal{G}(t)$. Means, both the nodes v_i and v_j are newly introduced in the network at time $t+1$.
3. Likewise, newly inserted nodes $v_i \in \mathcal{G}(t)$ and $v_j \notin \mathcal{G}(t)$.

For each incoming edge $e(v_i, v_j)$, we update the adjacency matrix accordingly and proceed as follows to handle the above scenarios.

Scenario 1: We compute membership function for both v_i and v_j with respect to C_i, when $v_i, v_j \in C_i$. Otherwise, compute $\mu_{C_i}(v_i)$ and $\mu_{C_j}(v_j)$, when $v_i \in C_i$ and $v_j \in C_j$. If we perceive a significant change between the previous and current membership scores, then the nodes are reassigned to a community with higher membership.

Scenario 2: A new community C_{new} is created containing both the nodes v_i and v_j.

Scenario 3: Membership of v_j with respect to C_i (since its associated node $v_i \in C_i$) is computed. If $\mu_{C_i}(v_j) \geq \eta$ then $v_j \in C_i$, otherwise a new community C_{new} is formed containing node v_j and recompute membership of v_i with respect to only those communities where neighbors of v_i are present. If needed reassignment of node v_i is performed.

To keep track of all the changes after every insertion of an edge, we use a membership matrix $M_C[\ ][\ ]$. The rows represent the total number of nodes present at time t and columns portray the number of neighbors along with the membership scores of a node concerning all existing communities at time t. After every new insertion of a node in the network, we allocate a new entry to M_C and update the neighbors' record of v_i. Simultaneously, we update the cluster feature vector. The steps of InOvIn is depicted in Algorithm 1.

In line 17 & 18 of Algorithm 1, a node (v_i) is assigned to the upper boundary of both the communities (C_j & C_p) if the membership score difference is within the threshold ζ. Two communities C_j and C_p are merged together when number of common nodes present in the upper boundary of both the communities are more than τ. Algorithm 2 describes how to merge (Merge()) for two communities C_j and C_p.

Due to the continuous insertion of nodes or edges to the network, the connectivity strength of some of the nodes gets affected. As a result, at $t+1$, a node may achieve a higher membership score with other communities. Consequently, the node will move from one community to another when there is a sufficient difference in membership score with the current

Algorithm 1: InOvIn: A rough-fuzzy based incremental community detection algorithm

Data: D ={$e_1(v_1, v_2), e_2(v_3, v_4), \cdots$} (Incremental network); ζ (Overlapping threshold)
Result: C = {$C_1, C_2, \cdots, C_m$} (Set of communities at time t)

1 Initialize C = {ϕ};
2 initialize $M_C[\ ][\ ] = \{0\}$;
3 Create Community $C_i = C_i \bigcup \{e_1\}$; D = D- {$e_1$}; C = C $\bigcup C_i$;
4 **for** $\forall e_j \in$ D **do**
5 **for** $\forall C_j \in$ C **do**
6 Compute $k = \max_j\{\text{Membership}(v_i, C_j)\}$; /* find the community (k) with maximum membership score of v_i */
7 **if** $(k = \phi)$ **then**
8 Create Community C_{new};
9 $C_{new} = C_{new} \bigcup v_i$;
10 C = C $\bigcup C_{new}$;
11 Update CF_C;
12 Update M_C;
13 **end**
14 **else if** $(|\mu_{C_j}(v_i) - \mu_{C_p}(v_i)| < \zeta)$/* $\forall C_p \in C$ and $j \neq p$ */
15 **then**
16 $\overline{C_j} = \overline{C_j} \bigcup \{v_i\}$;
17 $\overline{C_p} = \overline{C_p} \bigcup \{v_i\}$; /* Assign to the upper bound of both the communities */
18 Merge(C_j, C_p);
19 Update CF_{C_j};
20 Update CF_{C_p};
21 Update M_C;
22 **end**
23 **else**
24 $C_j = C_j \bigcup \{v_i\}$;
25 Update CF_{C_j};
26 Update M_C;
27 **end**
28 Periodically monitor M_C matrix ;
29 Compute $C_k = \max_j(\mu_{C_j}(v_i))$; /* Using M_c matrix */
30 Shift v_i to C_k ;
31 Update CF_C;
32 Update M_C;
33 **end**
34 **end**
35 Return(C) ;

Algorithm 2: Merge(C_j, C_p)

Data: N ={$v_1, v_2, \cdots$} (Nodes present in upper approximation of any two communities)
Result: C = {$C_1, C_2, \cdots, C_m$} (Set of community at time t) ; τ (Merging threshold)

1 **for** $\forall C_j \in$ C **do**
2 **for** $\forall N \in \overline{C_p}$ **do**
3 **if** $(|N| \geq \tau)$ **then**
4 Merge (C_j, C_p);
5 C = C $- \{C_p\}$;
6 Update CF_{C_j};
7 Update M_C;
8 **end**
9 **else**
10 $N = C_j \cap C_p$; /* Overlapping points */
11 **end**
12 **end**
13 **end**
14 Return(C) ;

Table 2
Input networks used for experimentations.

Network	Dataset	Available	#Nodes	#Edges	#Communities	#Intrinsic	#Overlapping
Synthetic	SyNet1	GitHub[2]	23	96	1	1	0
	SyNet2		48	121	5	0	5
	SyNet3		31	118	1	2	0
	SyNet4		63	133	1	1	0
	SyNet5		34	94	2	1	3
	SyNet6		43	140	2	2	3
Real-World	Karate network	UCI[3]	34	78	2	–	2 [40]
	American College Football		115	616	11	–	7 [41]
	Dolphin Social Network		62	159	2	–	2 [40]
	Books about US Politics		105	441	3	–	3 [42]
	Les Miserables		76	254	7	–	7 [41]
	Word Adjacencies		112	425	1	–	–
	Arxiv General Relativity (ca-GrQc)	SNAP[4]	5242	14,496	–	–	–
	Arxiv High Energy Physics Theory (ca-HepTh)		9877	25,998	–	–	–
	Social Circles from Facebook		4039	88,234	–	–	–
	Email-Enron		36,692	183,831	–	–	–
	ca-CondMat		23,133	93,497	–	–	–
	ca-HepPh		12,008	118,521	–	–	–

– Information not available.
Note: All these synthetic and real-world networks used in the experiment are undirected.

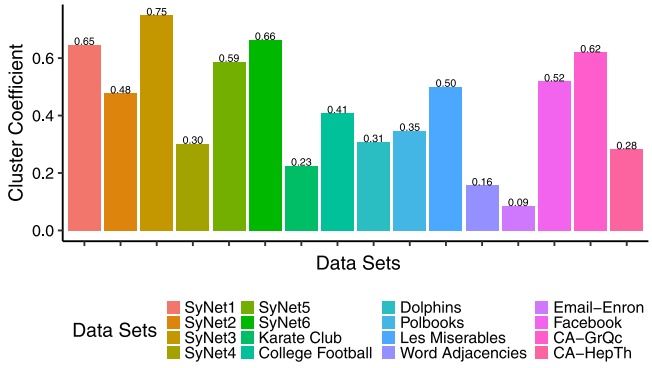


Fig. 5. Clustering Coefficients of different networks used.

community. To detect the possible formation of any embedded community, each community feature vector is examined within a certain time interval. We initiate the process by taking the node with the highest neighbors (degree). In line 3 of Algorithm 3, we are checking density variation using Eq. (5). A new embedded community C_{new} within C_j is created only when a node v_j satisfies the necessary condition to form a new community (line 5). Even so, after detection of density variation within a community a new community may be formed only when there is a minimum number of neighbors. We determine the minimum neighbors by adding mean and standard deviation of the current degree distribution i.e. $\overline{CF}_{C_i} + \sigma^{(t+1)}(CF_{C_i})$. We expand further the neighbors of v_j to detect an embedded community. The community expansion module is given in Algorithm 4.

In the next section, we evaluate the performance of `InOvIn` using several real and synthetic networks and compare its performance with contemporary community detectors.

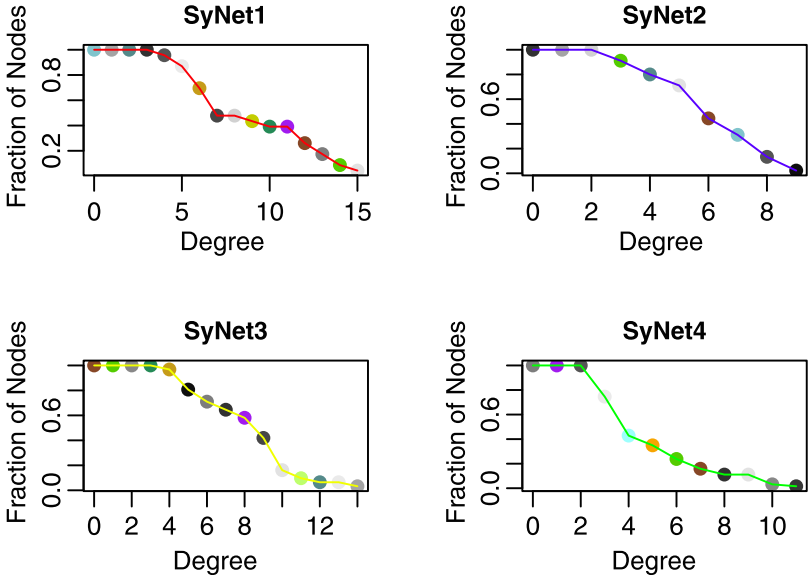


Fig. 6. Degree distribution of SyNet1, SyNet2, SyNet3 and SyNet4 showing power-law distribution trends.

Algorithm 3: Embedded Community

Data: CF ={$CF_{C_1}, CF_{C_2}, \cdots CF_{C_m}$} (Set of Community Feature Vectors at time t+1)
Result: C (Set of community at time t+1)

for $\forall CF_{C_i} \in$ CF **do**
  Compute $\max_j\{\delta_j \in CF_{C_i}\}$;
  /* find the maximum degree node in C_i */
  if $\overline{CF}_{C_i} > (\delta_j - \xi * \sigma^t(CF_{C_i}))$ /* Checking Density variation */
  then
    if $\delta_j > (\overline{CF}_{C_i} + \sigma^{(t+1)}(CF_{C_i}))$ /* to check minimum neighbor of v_j to form new community */
    then
      Create Community C_{new};
      $C_{new} = C_{new} \bigcup v_j$;
      for $\forall v_k \in Neighbors_\theta(v_j)$ /* Expanding the neighbor of v_j */
      do
        if $\delta_k > (\delta_j - \xi * \sigma^{(t+1)}(CF_{C_i}))$ **then**
          $C_{new} = C_{new} \bigcup v_k$;
          ExpandCommunity (v_k);
        end
      end
    end
  end
  Update $CF_{C_{new}}$;
  Update CF_{C_i};
  C = C $\bigcup C_{new}$;
end
Return(C) ;

Algorithm 4: ExpandCommunity (v_k)

for $\forall v_p \in Neighbors_\theta(v_k)$ /* Expanding the neighbor of v_k */
do
  if $\delta_p > (\delta_j - \xi * \sigma^{(t+1)}(CF_{C_i}))$ **then**
    $C_{new} = C_{new} \bigcup v_p$;
    ExpandCommunity (v_p);
  end
end

5. Performance evaluation

We implement InOvIn, in R and the executable code is available at.[1] We use six (06) synthetic and twelve (12) real-world networks for evaluation of our model. We compare the performance of InOvIn with nine (09) contemporary community detection methods, namely COPRA [25], EAGLE [13], SLPAD [11], AFOCS [28], OSLOM [30], Infomap [17], DyPerm [14], TILES [29] and HAM [18]. We appraise the performance of candidate methods based on ten (10) different assessment matrices. All the experiments are run in Windows 7 (64-bit) environment, having a machine configuration of 6 GB RAM and 2.70 GHz processor. To avoid implementation bias and for a fair comparison, we use publicly available code for the baseline methods. However, for TILES, LPANNI, DyPerm and HAM we could not find any such publicly available implementations. Hence, we decide to use the reported assessment scores for the comparison. We observe that NMI is the only common assessment index that most of the methods used for comparison and reporting. Accordingly, we too use NMI for comparing all the candidate methods. We use both static and dynamic networks for detecting different type of communities. Due to unavailability of networks with overlapping and intrinsic communities, we generate synthetic networks and report the input networks that we used for experimentations, next.

[1] https://github.com/nathkeshab/InOvIn.

5.1. Synthetic network generation

We generate six Synthetic Networks **SyNet** (1 to 6) with intrinsic communities (Please *ref* Sec.5.1, Fig. 3 in the supplementary material). The synthetic networks **SyNet (1 to 6)** contains three types of communities: disjoint, overlapping and intrinsic. Both SyNet1 and SyNet4 networks contain one intrinsic community, whereas the SyNet3 network contains two intrinsic communities. SyNet2 is created with five disjoint communities and two overlapping nodes. Networks SyNet5 and SyNet6 have all three kinds (disjoint, overlapping and intrinsic) of communities. **SyNet** networks are available at GitHub[2] for download.

5.2. Real networks used

We also use well known real-world networks, which are available at UCI.[3] machine repository and SNAP[4] A brief description of both synthetic and real-world networks used in the experiment are presented in Table 2. Usually, the total number of communities for most of the benchmark real-world networks are missing. We, therefore, consider the total number of communities as reported by the majority of the prior research work. In the case of the intrinsic community, we do not find any reported evidence of the presence of intrinsic communities in real-world networks.

5.3. Characteristic of input networks

We calculate clustering coefficient (CC), degree distribution (DD) (cumulative) and average path length (APL) of the input networks as demonstrated in Figs. 5, 6, 7 and 8 respectively. Computed values imply that all the **SyNet** follow properties of scale free networks with higher clustering coefficient and low average path length. The degree distribution of all the **SyNet** networks follows the power law property.

5.4. Experimenting with static networks

We consider above input networks (synthetic and real) as static and assess the quality of the detected communities using well-know assessment parameters. The color representation of the communities in two-dimensional space detected by a method is one of the easiest ways to access the quality of community detection. At first, we represent different communities in different color codes detected by InOvIn and report in the supplementary material (*ref* Sec. 5.4, Fig. 4 and 5). From the figure, it can easily be observed that InOvIn is effective in detecting both overlapping and intrinsic communities.

For qualitative evaluation of different community detectors, we consider various validity measures (including some overlapping validity measures) and compared them with other well-established community detectors. Next, we discuss the qualitative performance of the methods.

5.4.1. Qualitative assessment of static communities

In our experiments, we consider few overlapping assessment measures, namely, Overlapping Normalized Mutual Information (ONMI) [43], Average F-Score (AFS) [44] and Omega Index (OI) [45], Adjusted Rand Index (ARI) [46], Extended Modularity (EQ) [13] and overlapping modularity Q_{ov}[25]. Moreover, we also consider some of the internal validity measurements like Dunn [47], Silhouette [48], Connectivity [49] and Davies–Bouldin (DB) [50]. In case of synthetic networks InOvIn outperformed all other

[2] https://github.com/nathkeshab/SyNet.
[3] http://archive.ics.uci.edu/ml/datasets.html.
[4] https://snap.stanford.edu/data/#citnets.

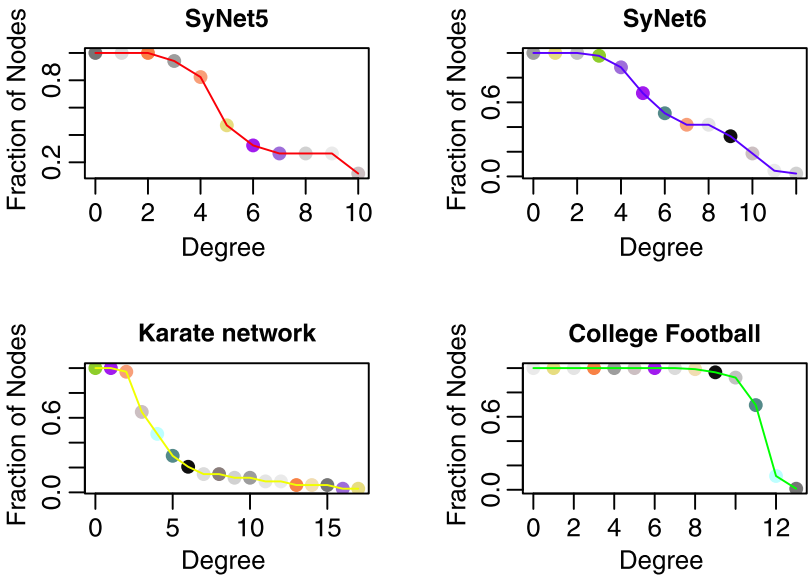


Fig. 7. Degree distribution of SyNet5, SyNet6, Karate network and American College Football showing power-law distribution trends.

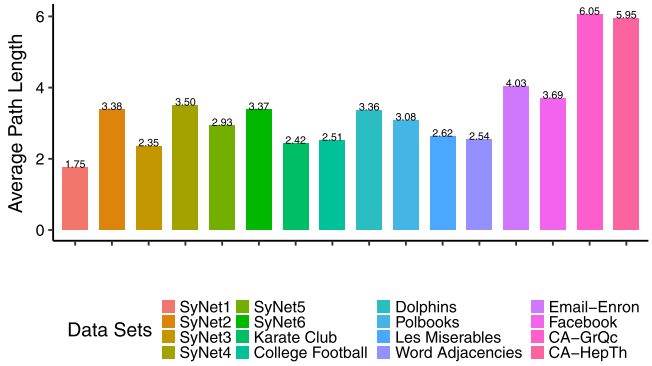


Fig. 8. Average path length of both synthetic and real world networks.

algorithms in terms of ONMI, AFS, Omega and EQ index (*see* Fig. 9) on networks SyNet2 and SyNet3. On SyNet1 network (*see* Fig. 9), Infomap marginally outperform `InOvIn` by about 1.02% in Omega index. EAGLE and COPRA outperform `InOvIn` by about 1.04% and 1.12% respectively based on EQ index on SyNet4.

In the case of real-world networks, `InOvIn` performs better on the karate network in comparison to other algorithms. On the dolphin network, our model achieves superior results concerning most of the assessment parameters, except the Omega index. In case of polbooks network, (*see* Fig. 10), only COPRA outperforms `InOvIn` with a marginal improvement of 1.34% based on Omega index. A very marginal performance difference of 0.06% is also obtained between `InOvIn` and CORPA in the Omega index on the Les Miserables network. On word adjacencies network (*see* Fig. 10), AFOCS produces similar results like `InOvIn` with a performance difference of just about 0.04% in the Omega index. CORPA and EAGLE outperformed `InOvIn` only in the EQ index by about 0.96% and 1.21% respectively on the word adjacency network. For more results on various synthetic and real-world networks, one can refer to the supplementary material (see Sec. 5.4.1, Fig. 7).

The quantitative assessment of `InOvIn` based on different internal validity measures [51] on both synthetic and real networks is presented in the supplementary material (*ref* Sec. 5.4.1, Table. 1 in the supplementary material). On real-world networks like dolphin, football and polbooks, `InOvIn` achieve satisfactory results (*see* Sec. 5.4.1, Fig. 6 in the supplementary material) than other algorithms for Dunn, Silhouette, and DB. By dissecting the results generated by `InOvIn`, it is evident that the performance of our approach is quite satisfactory and efficient concerning various validity measures.

Performance of `InOvIn` in static scenarios is assessed based on Normalized Mutual Information (NMI) [40] score. NMI is a measure for comparing similarity among communities. Its value lies between 0 and 1, with 1 corresponding to high similarity. The NMI scores for different methods are compared and reported in Fig. 11. Note that, we consider the NMI score as reported in the respective research articles. The comparative scores confirm the fact that our method is equally effective in detecting static communities.

5.4.2. Evaluation of overlapping static communities

We also consider Normalized Fuzzy Mutual Information (NFMI) [20] index, which is suitable for assessing the fuzzy community structures. Higher the value of NFMI, better is the quality of the detected community structures. NFMI compares the identified communities with the corresponding ground truth. In our experiment, we consider five real-world networks and process them incrementally. Initially, we exclude some of the overlapping edges from the networks and during the NFMI calculation, we add them incrementally. Fig. 12, shows that NFMI value decreases when the number of overlapping vertices increases in all the instances.

Since, the traditional modularity index [52,53] is designed to measure only the quality of disjoint communities, hence, we consider a new modularity measure proposed by Nicosia et al. [54], which is a variant of original modularity index for overlapping communities. We use this overlap modularity (Q_{ov}) index for the experimentation. Results obtain from the four algorithms on large real-world networks are presented in Fig. 13. It concedes that `InOvIn` performs better on networks *Facebook, ca-HepPh* and *Email-Enron* in comparison to other methods such as OSLOM, iLCD and COPRA. On the other hand, LPANNI outperform `InOvIn` based on Q_{ov} score.

5.5. Experimenting with evolving networks

Due to the lack of evolving networks, we simulate such networks for experimentation. To simulate evolving networks, we

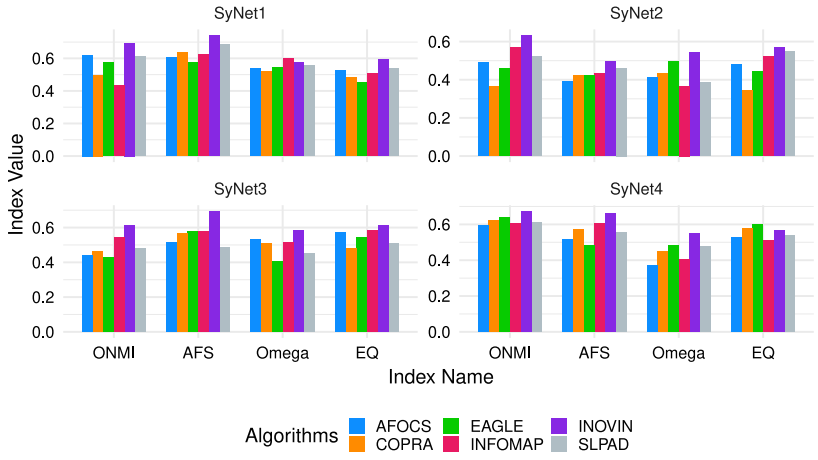


Fig. 9. Performance of different methods on **SyNet** 1 to 4 on various assessment parameters that establish `InOvIn` as a better performer.

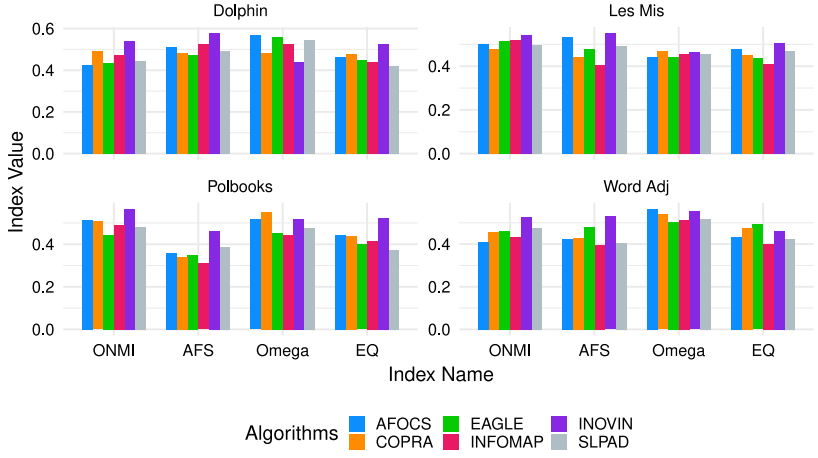


Fig. 10. Assessment of different detectors on Dolphin Social Network, Books about US Politics, Les Miserables and Word Adjacencies.

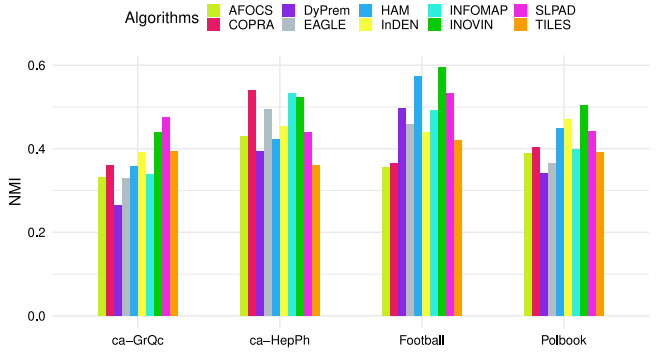


Fig. 11. NMI score obtained by different community detectors in static networks.

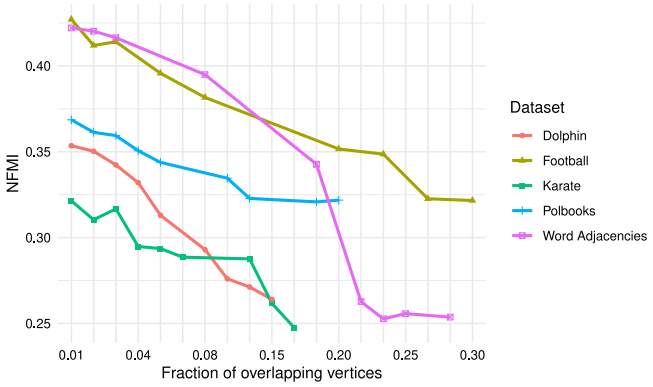


Fig. 12. NFMI performance of `InOnIn` on real datasets.

keep on adding a new subset of edges (un-processed) in a regular interval and compute an NMI score in each interval. This process is continued until all the edges of a network are completely processed. We use the synthetic network (SyNet6) and real-world networks-Polbooks, ca-CondMat and Facebook for the experimentation and report the outcomes in Fig. 14. Note that, in this experiment static algorithms like COPRA, EAGLE, and infomap are run through each snapshot independently. Results reported reveals that our approach produces comparatively better outcomes, both in synthetic and real networks in comparison to other algorithms. For more results on other networks, one can refer to the supplementary material (Sec. 5.4.1, Fig. 8).

Due to the continuous growth of a network, there is a possibility of community merging, expansion, node shifting, splitting with time. This fact may be established by looking into the number of communities extracted by `InOvIn` in different real-world networks with respect to time, shown in Fig. 15. It further confirm that the number of communities fluctuates with time.

Experimentally we are able to detect the existence of intrinsic communities in two real-world datasets namely *PolBooks* and

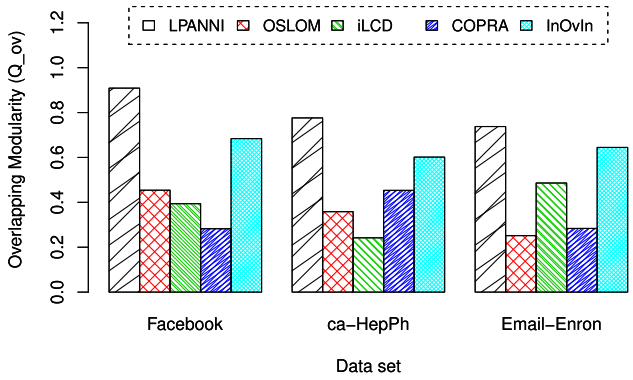


Fig. 13. Overlapping modularity (Q_{ov}) performance of four algorithms on large real datasets.

Word Adjacencies. *PolBooks* network contains two intrinsic communities of size 20 and 24, whereas *Word Adjacencies* network shows a single intrinsic community of size 47. To the best of our knowledge, no prior community detection algorithms report the presence of intrinsic communities in these two networks. From our experiment, we observe that the value of ξ of 2.1 achieves superior results in detecting intrinsic communities. However, a possible range of value for ξ may vary from 1.9 to 3, depends on the network under consideration.

5.5.1. Scalability of *InOvIn* in handling evolving networks

To test the scalability of `InOvIn` for handling growing networks, we record the execution time for both non-incremental and incremental scenarios. We start with a dataset of size 1000 and gradually increases the size of the network up to 25,000. In a non-incremental scenario, we report the total time requirements in seconds by re-running our method with varying network sizes from scratch. To simulate an incremental scenario, we treat the network as evolving and report the execution time after a certain interval. The execution time of `InOvIn` with varying network size is shown in Fig. 16. The results give a clear indication that `InOvIn` is scalable to the growing network size. Our approach is faster by a factor of approximately 13%, 23% and 17% than traditional non-incremental approaches for the size 5000, 10,000, 20,000 respectively.

We further perform scalability comparison based on CPU running time with four traditional methods (iLCD, LabelRankT, OSLOM and InDEN) using networks Enron, Facebook, ca-GrQc and ca-HepTH. We record computational time at different network snapshots ($m/4$, $m/3$, $m/2$ and m), where m represent the total number of edges. We highlight the best scores of candidate methods in Table 3. Scalability comparison shows a satisfactory performance by `InOvIn`.

6. Conclusion

We introduced an incremental approach `InOvIn` for detecting overlapping and intrinsic community in evolving networks. A rough-fuzzy concept is used to cluster communities with varying

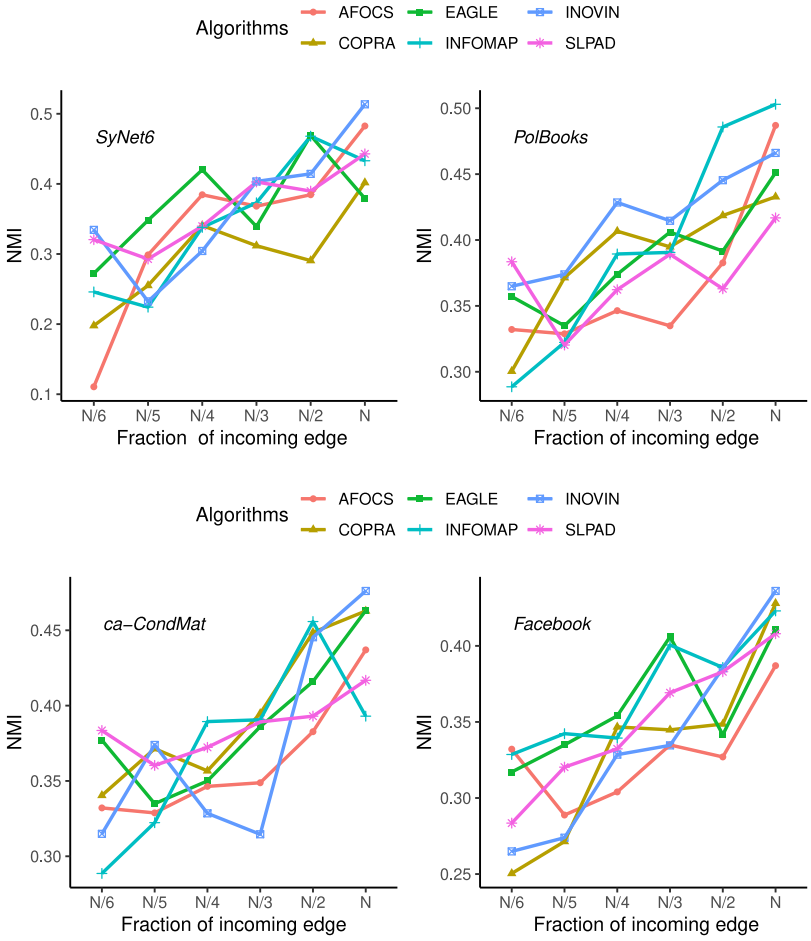


Fig. 14. Results generated by six algorithms on SyNet6 network, PolBooks network, ca-CondMat and Facebook with respect to NMI (Y-axis) and fraction of incoming edge (X-axis). **N** represent the size of the corresponding networks.

Table 3
Comparison of cumulative CPU running time (s).

Network		Existing Methods				**InOvIn**			
		$(m/4)$	$(m/3)$	$(m/2)$	(m)	$(m/4)$	$(m/3)$	$(m/2)$	(m)
Email-Enron	iLCD	**109.78**	276.17	488.91	792.67	116.34	**259.83**	459.77	**783.34**
	LabelRankT	122.67	266.87	465.45	787.32				
	OSLOM	132.07	306.42	**415.13**	785.12				
	InDEN	119.93	296.02	505.23	807.24				
Facebook	iLCD	**93.76**	183.96	302.81	681.23	106.42	196.54	**297.96**	**679.95**
	LabelRankT	123.56	214.41	358.65	701.32				
	OSLOM	132.10	206.50	387.14	698.02				
	InDEN	102.67	**176.07**	318.95	691.52				
CA-GrQc	iLCD	101.34	188.87	271.45	572.06	**98.54**	187.78	268.89	**524.78**
	LabelRankT	114.98	197.61	272.70	558.34				
	OSLOM	108.17	214.59	305.15	607.63				
	InDEN	115.39	**181.32**	**265.15**	547.32				
CA-HepTh	iLCD	102.08	218.95	311.78	603.65	108.87	**202.39**	324.12	**596.76**
	LabelRankT	117.56	225.87	296.45	611.34				
	OSLOM	121.14	226.51	315.26	628.21				
	InDEN	**92.67**	206.87	**285.71**	631.65				
CA-CondMat	iLCD	132.21	**278.15**	461.08	803.54	138.43	282.92	442.26	**786.60**
	LabelRankT	137.05	295.24	**438.51**	791.13				
	OSLOM	141.41	316.51	503.62	810.16				
	InDEN	**128.03**	306.12	475.11	788.45				

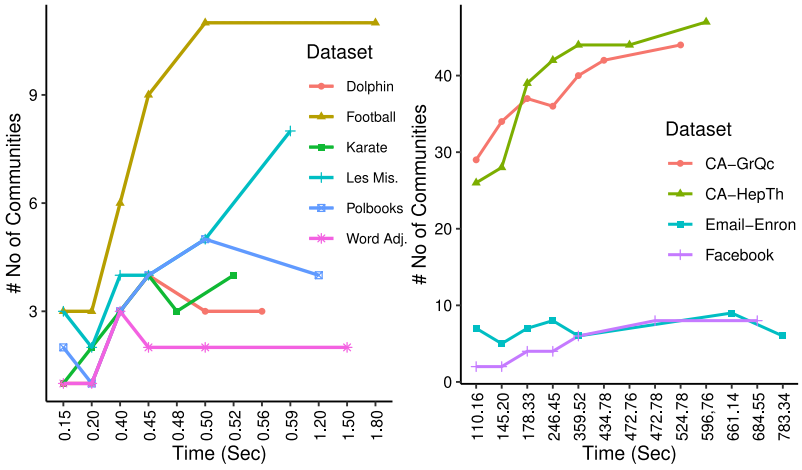


Fig. 15. Number of communities derived by InOvIn from both small (left) and large (right) input networks in different time intervals.

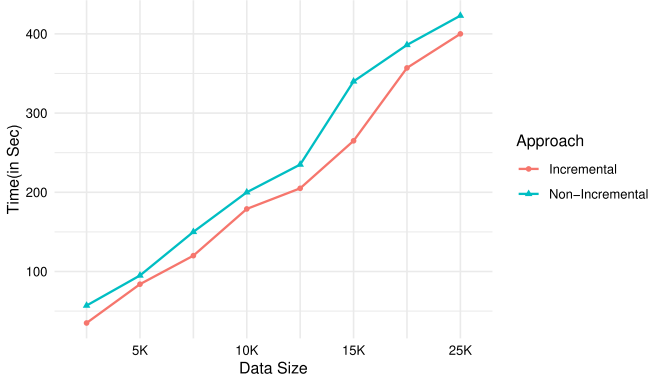


Fig. 16. Scalability of InOvIn in handling growing networks.

connectedness density. InOvIn used two different concepts. First, a fuzzy membership for measuring dynamic membership of an object towards a community and second, a density variation detection technique to detect embedded communities. We used both synthetic and real-world networks to access and compare the effectiveness of InOvIn. The quality of community detected by InOvIn is better than contemporary detectors for detecting intrinsic and overlapping communities in incremental networks.

We restricted our current study within evolving networks only and ignoring shrinking scenarios where nodes may become inactive over time. Work is on to derive a parallel version of InOvIn that can handle shrinking networks too.

Availability

R implementation of InOvIn and Synthetic networks used are available for download at https://github.com/nathkeshab/InOvIn and https://github.com/nathkeshab/SyNet.

Declaration of competing interest

No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.asoc.2020.106096.

CRediT authorship contribution statement

Keshab Nath: Data curation, Investigation, Methodology, Software, Validation, Visualization, Writing - original draft. **Swarup Roy:** Conceptualization, Formal analysis, Methodology, Project administration, Supervision, Writing - original draft, Writing - review & editing. **Sukumar Nandi:** Conceptualization, Writing - review & editing.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.asoc.2020.106096.

References

[1] H.N. Manners, S. Roy, J.K. Kalita, Intrinsic-overlapping co-expression module detection with application to Alzheimer's Disease, Comput. Biol. Chem. 77 (2018) 373–389.

[2] N.J. Krogan, G. Cagney, H. Yu, G. Zhong, X. Guo, A. Ignatchenko, J. Li, S. Pu, N. Datta, A.P. Tikuisis, et al., Global landscape of protein complexes in the yeast Saccharomyces cerevisiae, Nature 440 (7084) (2006) 637.

[3] A.-C. Gavin, P. Aloy, P. Grandi, R. Krause, M. Boesche, M. Marzioch, C. Rau, L.J. Jensen, S. Bastuck, B. Dümpelfeld, et al., Proteome survey reveals modularity of the yeast cell machinery, Nature 440 (7084) (2006) 631.

[4] H. Yu, P. Braun, M.A. Yıldırım, I. Lemmens, K. Venkatesan, J. Sahalie, T. Hirozane-Kishikawa, F. Gebreab, N. Li, N. Simonis, et al., High-quality binary protein interaction map of the yeast interactome network, Science 322 (5898) (2008) 104–110.

[5] E. Ravasz, A.L. Somera, D.A. Mongru, Z.N. Oltvai, A.-L. Barabási, Hierarchical organization of modularity in metabolic networks, Science 297 (5586) (2002) 1551–1555.

[6] R. Guimera, L.A. Amaral, Functional cartography of complex metabolic networks, Nature 433 (7028) (2005) 895.

[7] A.M. Feist, C.S. Henry, J.L. Reed, M. Krummenacker, A.R. Joyce, P.D. Karp, L.J. Broadbelt, V. Hatzimanikatis, B.Ø. Palsson, A genome-scale metabolic reconstruction for Escherichia coli K-12 MG1655 that accounts for 1260 ORFs and thermodynamic information, Mol. Syst. Biol. 3 (1) (2007) 121.

[8] G. Palla, A.-L. Barabási, T. Vicsek, Quantifying social group evolution, Nature 446 (7136) (2007) 664.

[9] M.C. Gonzalez, C.A. Hidalgo, A.-L. Barabasi, Understanding individual human mobility patterns, nature 453 (7196) (2008) 779.

[10] R. Cazabet, F. Amblard, C. Hanachi, Detection of overlapping communities in dynamical social networks, in: Social Computing (SocialCom), 2010 IEEE Second International Conference on, IEEE, 2010, pp. 309–314.

[11] N. Aston, J. Hertzler, W. Hu, et al., Overlapping community detection in dynamic networks, J. Softw. Eng. Appl. 7 (10) (2014) 872.

[12] Y.-Y. Ahn, J.P. Bagrow, S. Lehmann, Link communities reveal multiscale complexity in networks, Nature 466 (7307) (2010) 761–764.

[13] H. Shen, X. Cheng, K. Cai, M.-B. Hu, Detect overlapping and hierarchical community structure in networks, Physica A 388 (8) (2009) 1706–1712.

[14] P. Agarwal, R. Verma, A. Agarwal, T. Chakraborty, DyPerm: Maximizing permanence for dynamic community detection, in: Advances in Knowledge Discovery and Data Mining - 22nd Pacific-Asia Conference, PAKDD 2018, Melbourne, VIC, Australia, June 3–6, 2018, Proceedings, Part I, 2018, pp. 437–449, http://dx.doi.org/10.1007/978-3-319-93034-3_35, https://doi.org/10.1007/978-3-319-93034-3_35.

[15] J. Xie, B.K. Szymanski, X. Liu, Slpa: Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process, in: Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on, IEEE, 2011, pp. 344–349.

[16] J. Han, M. Kamber, J. Pei, Data Mining: Concepts and Techniques, Elsevier, 2011.

[17] M. Rosvall, C.T. Bergstrom, Maps of random walks on complex networks reveal community structure, Proc. Natl. Acad. Sci. 105 (4) (2008) 1118–1123.

[18] Y.-H. Fu, C.-Y. Huang, C.-T. Sun, A community detection algorithm using network topologies and rule-based hierarchical arc-merging strategies, PLoS One 12 (11) (2017) e0187603.

[19] F. Meng, F. Zhang, M. Zhu, Y. Xing, Z. Wang, J. Shi, Incremental density-based link clustering algorithm for community detection in dynamic networks, Math. Probl. Eng. 2016 (2016).

[20] S. Kundu, S.K. Pal, Fuzzy-rough community in social networks, Pattern Recognit. Lett. 67 (2015) 145–152.

[21] 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.

[22] N.B. Dillen, A. Chakraborty, Modularity-based community detection in fuzzy granular social networks, in: Proceedings of the International Congress on Information and Communication Technology, Springer, 2016, pp. 577–585.

[23] E. Becker, B. Robisson, C.E. Chapple, A. Guénoche, C. Brun, Multifunctional proteins revealed by overlapping clustering in protein interaction network, Bioinformatics 28 (1) (2012) 84–90.

[24] J. Sheng, K. Wang, Z. Sun, B. Wang, F. Khawaja, B. Lu, J. Zhang, Overlapping community detection via preferential learning model, Physica A 527 (2019) 121265.

[25] S. Gregory, Finding overlapping communities in networks by label propagation, New J. Phys. 12 (10) (2010) 103018.

[26] M. Lu, Z. Zhang, Z. Qu, Y. Kang, LPANNI: Overlapping community detection using label propagation in large-scale complex networks, IEEE Trans. Knowl. Data Eng. 31 (9) (2018) 1736–1749.

[27] S. Ding, Z. Yue, S. Yang, F. Niu, Y. Zhang, A novel trust model based overlapping community detection algorithm for social networks, IEEE Trans. Knowl. Data Eng. (2019) http://dx.doi.org/10.1109/TKDE.2019.2914201.

[28] N.P. Nguyen, T.N. Dinh, S. Tokala, M.T. Thai, Overlapping communities in dynamic networks: their detection and mobile applications, in: Proceedings of the 17th Annual International Conference on Mobile Computing and Networking, ACM, 2011, pp. 85–96.

[29] G. Rossetti, L. Pappalardo, D. Pedreschi, F. Giannotti, Tiles: an online algorithm for community discovery in dynamic social networks, Mach. Learn. 106 (8) (2017) 1213–1241.

[30] A. Lancichinetti, F. Radicchi, J.J. Ramasco, S. Fortunato, Finding statistically significant communities in networks, PLoS One 6 (4) (2011) e18961.

[31] M. Xu, Y. Li, R. Li, F. Zou, X. Gu, EADP: An extended adaptive density peaks clustering for overlapping community detection in social networks, Neurocomputing 337 (2019) 287–302.

[32] H. Wu, L. Gao, J. Dong, X. Yang, Detecting overlapping protein complexes by rough-fuzzy clustering in protein-protein interaction networks, PLoS One 9 (3) (2014) e91856.

[33] K. Nath, S. Roy, Detecting intrinsic communities in evolving networks, Soc. Netw. Anal. Min. 9 (1) (2019) 13.

[34] S. Gregory, An algorithm to find overlapping community structure in networks, in: European Conference on Principles of Data Mining and Knowledge Discovery, Springer, 2007, pp. 91–102.

[35] J. Xie, M. Chen, B.K. Szymanski, LabelRankT: Incremental community detection in dynamic networks via label propagation, 2013, CoRR abs/1305.2006, URL http://arxiv.org/abs/1305.2006.

[36] L.A. Zadeh, Fuzzy sets, Inf. Control 8 (3) (1965) 338–353.

[37] Z. Pawlak, Rough set theory and its applications to data analysis, Cybern. Syst. 29 (7) (1998) 661–688.

[38] A.-L. Barabási, R. Albert, H. Jeong, Scale-free characteristics of random networks: the topology of the world-wide web, Physica A 281 (1) (2000) 69–77.

[39] K. Nath, S. Roy, S. Nandi, Incremental approach for detecting arbitrary and embedded cluster structures, in: International Conference on Model and Data Engineering, Springer, 2016, pp. 220–233.

[40] A. Lancichinetti, S. Fortunato, J. Kertész, Detecting the overlapping and hierarchical community structure in complex networks, New J. Phys. 11 (3) (2009) 033015.

[41] D. He, D. Jin, Z. Chen, W. Zhang, Identification of hybrid node and link communities in complex networks, Sci. Rep. 5 (2015) 8638.

[42] X. Cao, X. Wang, D. Jin, Y. Cao, D. He, Identifying overlapping communities as well as hubs and outliers via nonnegative matrix factorization, Sci. Rep. 3 (2013) 2993.

[43] A.F. McDaid, D. Greene, N.J. Hurley, Normalized mutual information to evaluate overlapping community finding algorithms, 2011, CoRR abs/1110.2515, URL http://arxiv.org/abs/1110.2515.

[44] C. Manning, R. Prabhakar, S. Hinrich, Introduction to Information Retrieval, Vol. 1, Cambridge University Press, Cambridge, UK, 2008.

[45] L.M. Collins, C.W. Dent, Omega: A general formulation of the rand index of cluster recovery suitable for non-disjoint solutions, Multivariate Behav. Res. 23 (2) (1988) 231–242.

[46] L. Hubert, P. Arabie, Comparing partitions, J. Classif. 2 (1) (1985) 193–218.

[47] J.C. Dunn, Well-separated clusters and optimal fuzzy partitions, J. Cybern. 4 (1) (1974) 95–104.

[48] P.J. Rousseeuw, Silhouettes: a graphical aid to the interpretation and validation of cluster analysis, J. Comput. Appl. Math. 20 (1987) 53–65.

[49] J. Handl, J. Knowles, D.B. Kell, Computational cluster validation in post-genomic data analysis, Bioinformatics 21 (15) (2005) 3201–3212.

[50] D.L. Davies, D.W. Bouldin, A cluster separation measure, IEEE Trans. Pattern Anal. Mach. Intell. (2) (1979) 224–227.

[51] K. Nath, S. Roy, S. Nandi, Incremental approach for detecting arbitrary and embedded cluster structures, in: International Conference on Model and Data Engineering (MEDI), in: Lecture Notes in Computer Science, vol. 9893, Springer, 2016, pp. 220–233.
[52] M.E. Newman, M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E 69 (2) (2004) 026113.
[53] M.E. Newman, Modularity and community structure in networks, Proc. Natl. Acad. Sci. 103 (23) (2006) 8577–8582.
[54] V. Nicosia, G. Mangioni, V. Carchiolo, M. Malgeri, Extending the definition of modularity to directed graphs with overlapping communities, J. Stat. Mech. Theory Exp. 2009 (03) (2009) P03024.