Data & Knowledge Engineering 113 (2018) 155–170

Contents lists available at ScienceDirect

Data & Knowledge Engineering

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

A spreading activation-based label propagation algorithm for overlapping community detection in dynamic social networks

Mohammad Sattari, Kamran Zamanifar*

Department of Computer Engineering, University of Isfahan, Isfahan, Iran

ARTICLE INFO

Keywords:
Spreading activation
Label propagation algorithm
Overlapping community detection
Dynamic social networks

ABSTRACT

Community detection in temporal social networks is an increasingly challenging subject in network analysis. The Label Propagation Algorithm (LPA) is a simple and fast approach for community detection in dynamic networks. However, it tends to generate monster communities which decrease the accuracy of community detection, especially in dynamic social networks. In this paper, we propose a modified LPA, called Spreading Activation Label Propagation Algorithm in order to solve the problem. This method assigns a property, called activation value, to each label, where pairs (label name, activation value) are propagated by spreading activation process and the LPA. Furthermore, this algorithm uses two weighting algorithms, where each of them corresponds to one variation of the proposed method. Here, the variations of the proposed method and other available methods on real and synthetic networks are implemented. Experimental results on both real and synthetic networks show that all variations of the proposed method detect communities more accurately compared to the benchmark methods while they are slower than these methods.

1. Introduction

Discovering communities, one of the important structures of social networks, is certainly a matter of debate. Extracting this structure is not only suitable for advertising, but also for optimizing Internet services [1]. A community in a network is a set of densely connected nodes that contains weaker connections to other nodes of the network [2]. There are many approaches for detecting communities in social networks, among which label propagation algorithm (LPA) is the simple and time-efficient approach [3–5]. This algorithm initializes each node with a unique label, indicating the community it belongs to [3]. Then, in each iteration, it updates the label of a node to the label that appears most frequently among its neighbours [3]. Finally, the nodes with same label form a community. Label propagation algorithm can detect only disjoint communities while the communities in real networks overlap with each other [6]. A case in point being in DBLP network, where researchers may belong to separate communities based on distinct paper subjects.

Owing to only requiring local information [7], LPA-based methods [7–9] are suitable candidates to detect communities in dynamic social networks. However, in this method, one or more labels may over-propagate leading to the formation of the monster communities. This kind of communities decrease the accuracy of community detection based on the label propagation, especially in dynamic social networks. To overcome this problem, we propose a new label propagation method based on spreading activation process. Spreading activation is a searching process used in centrality measurement [10] and semantic networks [11]. In spreading activation, a few source nodes is randomly chosen, where each of them possesses an activation value. This value represents the strength of a source node for spreading activation out to its neighbours.

* Corresponding author.
E-mail addresses: mohammadsattari9220@eng.ui.ac.ir (M. Sattari), zamanifar@eng.ui.ac.ir (K. Zamanifar).

https://doi.org/10.1016/j.datak.2017.12.003
Received 30 November 2016; Received in revised form 23 September 2017; Accepted 24 December 2017
Available online 02 January 2018
0169-023X/

The proposed method, called Spreading Activation-based Label Propagation Algorithm (SALPA), initializes each node with a pair (label name, activation value) in the network. Next, it applies two graph weighting algorithms [12,13], where each of them corresponds to a variation of the proposed method. Then, each node sends its activation value to other nodes of the network based on breadth-first search until the activation value get below the threshold. Finally, each node updates or adds the pair (label name, activation value) based on the activation value of its neighbours. On the whole, the main contributions of this newly proposed method are threefold:

This paper proposes a new method based on label propagation algorithm and spreading activation process in order to improve community detection in overlapping dynamic social networks.

The proposed method extends label propagation algorithm by assigning the pairs (label name, activation value) instead of one label to each node.

We apply two weighting algorithms, forming separate variations of the proposed method.

The structure of this paper is as follows: we present the literature review in Section 2. Section 3 describes the notation and the spreading activation process. Section 4 outlines the proposed method and time complexity. In Section 5, the experiment results are discussed and finally, Section 6 concludes the paper.

2. Literature review

There are several community detection approaches in social networks such as game theory [14–16], seed expansion [17,18] modularity optimization [19,20], clustering [21,22], and label propagation algorithm [3].

In game theory approach [14–16], each node is a selfish agent that tries to maximize its utility by joining new communities, switching to another community, leaving its community, or taking no action. However, forming communities by this approach is NP-hard [16]. In seed expansion approach [17,18], communities are extracted based on seeds, which are dense small communities. Cazabet et al. [17] presented Intrinsic Longitudinal Community Detection (ILCD) which tried to find small communities growing gradually with the passage of time. Ma et al. [18] proposed the method that extracted full graphs three vertexes as seeds by a recursive algorithm. These seeds are updated by adding or removing nodes or edges. As a seed are much smaller than a community, the updating time of the seed is much lower than that of the community. However, this approach considered one special type of communities which decreases the accuracy of community detection.

In modularity optimization approach [19,23], communities are detected in order to maximizing modularity. Clauset [19] considered a sparse matrix for each pair of communities which there exists at least one edge between them. The time complexity of this method in sparse networks is low while the accuracy of that is also low. Nguyen et al. [20] proposed Quick Community Adaptation algorithm (QCA) for community detection. This method traced the community structure and applied various strategies based on nodes and edges added or removed. However, this method produces very small communities, which decrease the accuracy of community detection. In clustering approach [21,22], a cluster is same as a community. Lin et al. [21] provided a framework for the analyzing the evaluation of communities without considering the overlap between them in real networks. Lancichinetti et al. [22] presented Order Statistics Local Optimization Method (OSLOM) which detected overlapping and hierarchical communities based on the edge direction and weight. However, this method produced many singleton communities, which leads to reduced accuracy.

A simple and time-efficient approach is the LPA which discovers communities by exchanging labels [3–5]. The simplicity is because each node has one label that shared with its neighbours and the time-efficiency is because the number of iterations of this approach is approximately 5 iterations [3]. Raghavan et al. [3] initially applied this approach in the community detection field and later as a result of the simplicity and practicality of this approach, other fields, such as finding influential nodes [23] and feature extraction [24], also employed the approach. The comparison of label propagation algorithm and other approaches based on running time, simplicity and accuracy is in Table 1.

Gregory et al. [25] presented Community Overlap PRopagation Algorithm (COPRA), where each node can possess one or more labels with different belonging factors. However, this method detects inaccurate communities in high overlapping networks. Speaker Listener Propagation Algorithm (SLPA) [26] is another algorithm based on label propagation algorithm, where it detects overlapping communities by assigning a set of labels to each node. Each label has the property called count which represents the number of the label in that node. This method attains high speed and can execute in a parallel manner [27]. However, this method produces large communities in sparse networks.

The LPA applied in dynamic networks as static networks. This approach initialized a property for each label in snapshot 1 and updated that in snapshots 2 and onwards. Xie et al. provided LabelRankT [28] being a dynamic extension of LabelRank [29], where

Table 1
The comparison of the LPA and other approaches.

Approaches	Running time	Simplicity	Accuracy
Label propagation algorithm	Low	High	Average
Clustering	Average	Average	Low
Modularity optimization	Average	Average	Low
Game theory	High	Low	Average
Seed expansion	Low	Average	Low

each node has a property called label distribution vector propagating to its neighbours. The same label distribution vector, the same community. In this method, each node belongs to at most one community; therefore it cannot detect overlapping communities. Aston et al. provided SLPAD [8] which is a dynamic version of the SLPA method proposed in [24]. In this method, one node is chosen randomly as a receiver and its neighbours are considered as speaker sending a label to that node. Liu also provided DLPAE [9] which is a dynamic extension of dominant label propagation algorithm [30]. It assigned a property called belonging factor to each label, where one of the node labels is considered as the main community and other labels are regarded as the part of the secondary communities [9]. This method is able to detect overlapping and non-overlapping communities while the running time of that is high.

3. Preliminaries

In this section, firstly, we provide notations used in the article and then introduce spreading activation process briefly.

3.1. Notation

Generally, a network is represented by a graph G(V,E), where V is the set of vertices and E is the set of edges. We define a dynamic network SG as a set of sequential graphs SG={G_1, G_2, ...,G_t}, where each graph corresponds to one snapshot; For example, G_1 corresponds to snapshot 1 and symbol T represents the snapshot number starting with 1. In real networks, snapshots are taken from systems while they are generated in synthetic networks. In view of the fact that the proposed method and other available methods were provided for overlapping dynamic community detection, this kind of communities is taken into consideration. Overlapping dynamic community detection is the task of finding densely connected overlapping sub graphs in SG.

3.2. Spreading activation

The Spreading activation process is a searching method is begun by choosing randomly one or more nodes as the source nodes in the graph. Each node is initialized with an activation value, where that of the source nodes be one and that of the other nodes be zero. The activation value of each node represents the amount of activation of the node in order to propagate information. Each source node sends its activation value to other nodes by the breadth-first search approach. Next, the receiver nodes update their activation value based on three parameters:

1. The activation value of the source node
2. The weight of the link between the source node and the target node
3. A decay factor.

The decay factor is the parameter that represents the amount of spreading strength reducing for edges of the graph. This process halts when a node is reached from more than one path. Finally, each source node creates a tree that consist of the nodes that are accessible by it. On the whole, the activation value of node j is computed based on the activation value of node i considering i is the source node and j is the target node [11]:

$$A[j] = A[j] + \sum_{i \in neighbours(j)} (A[i]^* \ W[i, \ j]^* \ D) \tag{1}$$

where A [i] and A[j] are the activation values of nodes i and j respectively. Symbol D is the decay factor and W[i,j] represents the weight of the link between nodes i and j.

4. The proposed method

The proposed method consists of four parts. The first part is initialization, where each node is labelled with a pair (label name, activation value). In this pair, firstly, the label name is set to the node number and the activation value is set to one. The second part is graph weighting, where it applies two important weighting algorithms. The first one is Weighted Edge Random Walk-k Path (WERW-Kpath) [12] and the second one is Social Behavioural Information Diffusion Model (SBIDM) [13].

Each weighting algorithm corresponds to one variation of the proposed method, where the first variation called Random Walk Spreading Activation Label Propagation Algorithm (RWSALPA) and the second variation called Social Behavioural Spreading Activation Label Propagation Algorithm (SBSALPA). The RWSALPA variation initializes the weight of each edge by 1. Next, it simulates $\boldsymbol{p}$ random paths, where for each path, at most k edges gain weight. Therefore, parameter k determines the maximum length of each path [12]. Each node in SBSALPA variation is assigned a score to and then it gets information based from the best neighbour. The best neighbour is the neighbour with the highest score computed based on Holland's Hyperplane Defined Function (HDF) [31]. After determining the best neighbour, the edge between the node and its best neighbour is incremented by *statevalue* of that node which is computed by [12]:

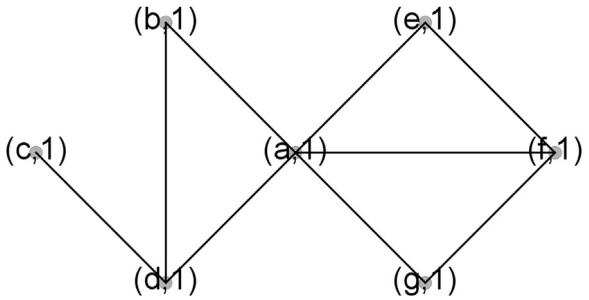


Fig. 1. The result of the first part of SBSALPA.

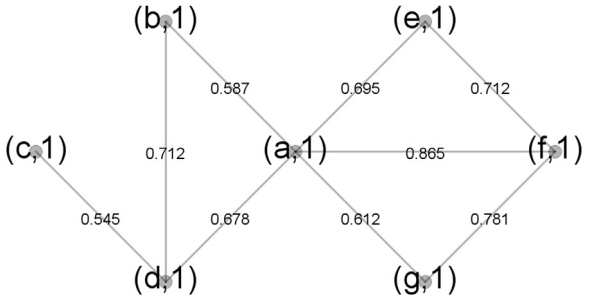


Fig. 2. The result of the second part of SBSALPA.

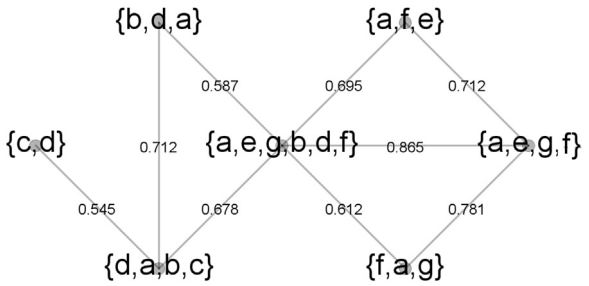


Fig. 3. The result of the third part of SBSALPA.

$$stateValue_v^{t+1} = \begin{Bmatrix} stateValue_v^t + neighbinf_v^{t+1} \\ stateValue_v^t + randNum \\ stateValue_v^t + neighbinf_v^{t+1} + randNum \end{Bmatrix} \quad (2)$$

where

$$neighbinf_v^{t+1} = neighbinf_v^t + \mathrm{a} * (bestNeighb_v^t - stateValue_v^t) \quad (3)$$

Both of weighting algorithms try to achieve the aim that the greater the edge weight, the greater the information flow. We applied SBSALPA in graph seven vertexes, where Figs. 1 and 2 show the result of the first and second part of SBSALPA on a graph seven vertexes.

We consider self-loop with pre-defined weight for all nodes in the third part (0.6 in Fig. 3). This part applies spreading activation process, where all new nodes instead of a few nodes are chosen as the source nodes. Next, each source node first sends its label activation value to itself and then send it to other nodes visited by breadth-first search until the goal state is reached. Here, the goal state is the state that the activation value get below the threshold. Each target nodes updates the activation value of the received label based on three parameters. The first parameter is the activation value of the source node, the second parameter is the weight of the edge between the source and target node and the third parameter is the decay factor. When the process finished, the labels with a low activation value are removed from the nodes. Fig. 3 show the result of the third part of SBSALPA in the graph.

In the fourth part, firstly, changed nodes are placed in a random order and afterwards, a node is selected from this order as the receiver node. Next, its neighbors are extracted and considered as the sender nodes. Then, each sender node chooses one of its labels whose activation value is greater than its other labels. After that, the sender nodes send their selected label to the receiver node and

afterwards, the receiver node compute the new activation value of the sender labels based on the weight of the edges between the sender and receiver nodes. The receiver node sums the activation value of the same labels and afterwards it accepts the label that sum of its activation values is greater than other labels.

If the activation value of the receiver label is greater than 1, the algorithm will set it to 1 as a result of the range of the activation value. Part 4 repeats between 5 and 20 times. Finally, labels with a low activation value are removed. Nodes with the approximately same activation value labels form communities. Fig. 4 show the result of the fourth part of SBSALPA in the graph. The results show that Nodes (a, e, f, g) and (a, b, c, d) are in the same community.

4.1. Algorithm description

Algorithm 4 presents the pseudo-code of SALPA. The algorithm first assigns the maximum activation value to all new nodes and afterwards, the edge weighting process is begun. For each snapshot, two weighting algorithms are applied by *WERW-KpathEdgeWeighting(Gts) or SBIDMEdgeWeighting(Gts)* functions, where *WERW-KpathEdgeWeighting(Gts)* corresponds to RWSALPA variation and *SBIDMEdgeWeighting(Gts)* corresponds to SBSALPA variation. These functions are presented in Algorithms 1 and 2. In Algorithm 1, first, the weight of all edges in the graph is initialized by 1. Next, a node v_n is randomly chosen as the stating node the path and one of its neighbors is selected as the next node by probability Pr. When the length of path reached k or all edges incident onto current node are traversed, loop "while" terminates. Finally, the function *Normalize (G(V,E))* adjusts the weight of the edges such that the weight of each edge is not more than one. Similar to Algorithm 1, Algorithm 2, first, initializes the weight of all edges by 1. Next, a node is randomly chosen and the score of its neighbours is computed. Among the neighbours, the node whose score is greater than other nodes selected as the best neighbour. Next, the weight between the node and its best neighbour is incremented by *statevalue* of that node. Similar to Algorithm 1, in Algorithm 2, the function *Normalize (G(V,E))* adjusts the weight of the edges such that the weight of each edge is not more than one.

In following, function *Spreading Activation (G(t))* which is in Algorithm 3 computes the activation value of the nodes. In this algorithm, first, for each node, an empty set called UnFiredNodeSet is created and then updated based on the breadth-first search. Nodes are extracted from this set and their activation value is computed based on Eq. (1) until the activation value get below than the threshold or UnFiredNodeSet be empty. Correspondingly in Algorithm 4, label propagation is begun. In this part, first, changed nodes are placed in random order by *ShuffleOrder()* function. Next, one of the nodes is chosen from this order as the receiver node and then function *ObtainNbs()* extracts neighbors of the receiver node as sender nodes. Then, *GetLabels()* function chooses one of the labels of each sender node that activation value is maximum and then it creates the candidate label set of the sender labels. After that, the receiver node accepts one of the labels that sum of its activation value is greatest among candidate set labels. Finally, labels with a low activation value are removed and communities are extracted based on similarity in the activation values of the labels.

Algorithm 1. WERW-KpathEdgeWeighting(G(t)) [12]

Input: (G = (V,E), k: an integer, ***p***:an integer)
Output: WG: weighted graph
Method:
1. **for each** $e_m \in E$
2. set $\omega(e_m)=1$
3. **end for each**
4. **for** i:=1 to ***p*** do
5. N=0; // a counter to check the length of k-path
6. Choose randomly v_n
7. **while** $N < k$ and $\left[|I(v_n)| < \sum_{e_k \in I(v_n)} T(e_k)\right]$ do
8. $e_m = e_m \in \{I(v_n) | T(e_m)=0\}$, chosen with probability Pr computed as following
$$\Pr(e_m)=\frac{\omega(em)}{\sum_{e_k \in \hat{I}(v_n)} T(e_k)}$$
9. Let v_{n+1} be the vertex reached by v_n through e_m
10. $\omega(e_m)= \omega(e_m)+1$
11. $T(e_m)=1$
12. $v_n = v_{n+1}$
13. N=N+1
14. **end while**
15. **end for**
16. WG=Normalize (G(V,E))

Algorithm 2. SBIDMEdgeWeighting(G_{ts}) [13]

Input: G = (V,E)
Output: WG: weighted graph
Method:
for each $e_m \in E$
  set $\omega(e_m)=1$
end for each
for i:=1 to n **do**
  Choose randomly v
  $bestNeighB_v = u \in neighbours\ of\ v$
  For $u \in neighbours\ of$ v do
    if$(score(stateValue_u^t) > score(bestNeighB_v)$ *then*
      $bestNeighB_v = u$
    end if
  end for
  $InterractionList_v$ = add($bestNeighb_v$)
  if$(score(stateValue_v^t) < score(bestNeighB_v)$ *then*
    Update $neighbinf_v^{t+1}$ based on Eq. (3)
    Update $stateValue_v^{t+1}$ based on Eq. (2)
    end if
  WE(v, $bestNeighB_v$)= WE(v, $bestNeighB_v$)+$stateValue_v^{t+1}$
end for
WG=Normalize (G(V,E))

Algorithm 3. Spreading Activation (G(t), new nodes)

Input: G(V,E)
Output: $G(V^+,E)$
Method:
for $v \in$ nodes do
  Create empty set UnfiredNodeSet
  Add v to UnFiredNodeSet
  Neighbours=v.ObtainNbs();
  Add Neighbours to UnFiredNodeSet
  while UnFiredNodeSet is not empty do
    Extract nv from UnFiredNodeSet
    Compute nv.label.activationvalue based on Eq. (1)
    If nv.label.activationvalue < threshold
      continue;
    end if
  Neighbours=nv.ObtainNbs();
  Add Neighbours to UnFiredNodeSet
end while
end for

Algorithm 4. SALPA

Input: SG={G_1= < V_1, E_1 > , G_2= < V_2, E_2 > ,..., G_n= < V_T, E_T > }, T
Output: set of communities of G_n
Method:

```
1.   ΔV = {v|v∈G1}
2.   for ts =1 : T do // ts stands for timestamp
3.     or v∈ΔV
4.        Node(v).label=v;
5.        Node(v).label.activationvalue=1
6.     end for
7.   ΔV = {vv∈G_ts+1 ∩ v∉G_ts} ts ≠ T
8.   end for
9.   ΔE = {ee∈G1}
10.  ΔV={v|v∈G1}
11.  for ts =1 : T do
     // Part 2 : Edge Weighting
12.       WERW-KpathEdgeWeighting(Gts) or SBIDMEdgeWeighting(Gts)
     // end Part 2
     // Part 3 : Spreading Activation
13.  Spreading Activation (G(t));
     // end Part 3
     // Part 4 : label propagation
14.       ChangedNodes={u,v|(u,v)∈ΔE}
15.       Vold={v | v∈ Gts∩ v ∉ G(ts+1) } }
16.       ChangedNodes= ChangedNodes -Vold
17.       for it = 1 : IT do // it means iteration
18.         ChangedNodes.ShuffleOrder();
19.         for i = 1 : ChangedNodes.count do
20.              Receiver=ChangedNodes(i);
21.              Senders= ChangedNodes(i).ObtainNbs();
22.              Candidatelabels=Senders.GetLabels();
23.              Receiver.update(CandiadateLabels.TheMostActivationValueLabel);
24.       end for
25.    end for
26.    remove Nodes(i) labels seen with activation value < r
27.    ΔE = {ee∈G_ts+1 ∩ e∉G_ts}∪{{ee∈G_ts ∩ e∉G_ts+1} ts ≠ T
28.    ΔV = {vv∈G_ts+1 ∩ v∉G_ts} ts ≠ T
       // end Part 4
29.  end for
```

4.2. Time complexity

The proposed method is divided into four parts. In part 1, it requires *O(n)* for initializing labels, where *n* is the number of nodes. In part 2, two weighting algorithms are applied. This part requires *O(m)* in the worst time, where *m* is the number of edges. In part 3,

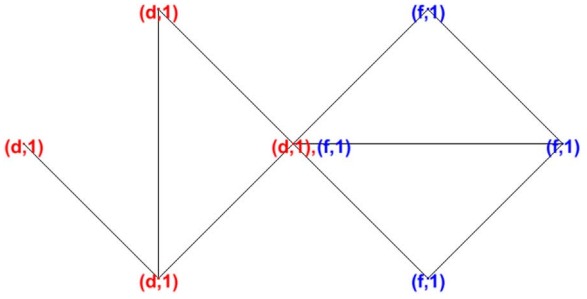


Fig. 4. The result of the fourth part of SBSALPA.

as we use breadth-first search for each node of the graph, it totally requires $O(n*m)$. Moreover, the part 4 including send and receiving labels requires $O(n)$. From the above fact, it can be concluded that the proposed method in each snapshot requires $O(n*m)$. Since the number of snapshots is a constant T, the overall complexity of community detection for all snapshots is $O(n*m)$.

5. Experiments

The performance of all variations of the newly proposed method and other earlier available methods are evaluated using four real and four synthetic networks.

5.1. Datasets description

We describe separately four real and four synthetic datasets in this part.

5.1.1. Real networks description

Real networks consist of four networks, where the first and the second are citation networks, the third is the email network and the fourth is the bibliography network. In the citation networks, nodes can only be added; therefore the number of nodes in comparison with the immediately preceding snapshot is increased. In this paper, directed edges are considered as undirected edges.

5.1.1.1. Arxiv HEP-PH [32]. Arxiv HEP-PH is a citation network covering articles published between January 1993 and April 2003. It contains 34,546 nodes and 421,578 edges, where the nodes are papers and the edges are citations between papers. Therefore, an edge from node i to node j is regarded as a citation between papers i and j. This dataset considered papers and communication of one month as one snapshot.

5.1.1.2. Arxiv HEP-TH [33]. Arxiv HEP-PH is a citation graph covering high-energy physics theory publications. As with the earlier dataset, papers are considered as nodes and citations between them are considered as the edges of the graph. It contains 27,770 nodes and 352,807 edges. This dataset consists of papers from an 11-year period from January 1993 to April 2003. This dataset considered papers and communication of two months as one snapshot. The summary of citation datasets is presented in Table 2.

5.1.1.3. Enron Email [34]. The email network used in this paper is the Enron Email dataset. This dataset incorporates email exchanged via Enron Email over 15-years, with each two month period being one snapshot. In this network, nodes are Enron employees and edges are emails exchanged between them.

5.1.1.4. DBLP [35]. DBLP is a computer science bibliography providing a comprehensive list of research papers in computer science. In this dataset, a co-authorship network is constructed, where two authors are connected if they publish at least one paper together. Unlike the citation networks, the DBLP network includes ground-truth communities. It considered papers published in ten years between 2000 and 2009. This dataset contains 317,080 nodes and 1,049,866 edges, where every ten days is considered as one snapshot. The DBLP dataset is similar to the Enron Email dataset in respect of supporting overlap communities. The summary of the DBLP and Enron Email datasets is presented in Table 3.

Table 2
Citation datasets summary.

Data	Arxiv HEP-PH	Arxiv HEP-TH
Number of nodes	34546	27770
Number of edges	421578	352807
Number of snapshots	124	62
Type	Directed, Temporal, Unweighted	Directed, Temporal, Unweighted

Table 3
Enron Email and DBLP dataset summary.

Data	Enron Email	DBLP
Number of nodes	36692	317080
Number of edges	367662	1049866
Number of snapshots	90	365
Type	Directed, Temporal, Unweighted	UnDirected, Temporal, Unweighted

Table 4
The core parameters used for the generation of synthetic networks.

S	n	avgd	Maxd	minc	maxc	O_n	O_{nc}	μ
10	7000	14	40	25	45	200	4	0.3

Table 5
Custom parameters used for the generation of synthetic networks.

Birth/Death	Birth/Expand	Expand/Contract	Expand/Death
20	20	20	20

5.1.2. Synthetic networks description

The performance of all variations of the proposed method and earlier available methods is evaluated in four synthetic networks. These networks are generated through methods introduced by [36]. The core parameters for generating these networks are presented in Table 4. In this table, symbol *s* is the number of snapshots and symbol *n* denotes the number of nodes. Symbols *avgd* and *maxd* represent the average and maximum degree of nodes respectively. Symbols *minc* and *maxc* represent the maximum and minimum size of the communities. Symbols O_n and O_{nc} specify the number of overlapping nodes and the number of communities to which each overlapping node belongs respectively. Symbol μ is the mixing parameter set to 0.3.

The custom parameters for generating four dynamic networks are presented in Table 5. In this table, each pair event is relevant to a dynamic generated network. We describe each event briefly in following:

Birth: a community forms at *i*th snapshot, if it not exists in the previous snapshots [37].

Death: a community dies at *i*th snapshot, if it not exists in the next snapshots [37].

Expand: a community expands at *i*th snapshot, if its number of nodes in that snapshot are increased [37].

Contract: a community contracts at *i*th snapshot, if its number of nodes in that snapshot are decreased [37].

Each parameter specifies the number of corresponding events per snapshot. For example, in the first dynamic network, twenty birth or death events occurred in each snapshot.

5.2. Evaluation

This paper applies the modularity measure for accuracy evaluation as a result of the lack of ground-truth communities in real networks [20]. The original modularity handles non-overlapping communities [2] while in this paper, communities can share members. We adopt the extended version of modularity [38] that can deal with the overlapping community structures as well:

$$Q_{Ov}^{c} = \frac{1}{2m} \sum_{c \in C} \sum_{i,j \in V} \left[A_{i,j} - \frac{k_i k_j}{2m} \right] \beta_{ic} \beta_{jc} \tag{4}$$

where

$$\beta_{ic} = \frac{k_{ic}}{\sum_{c'} k_{ic'}} \tag{5}$$

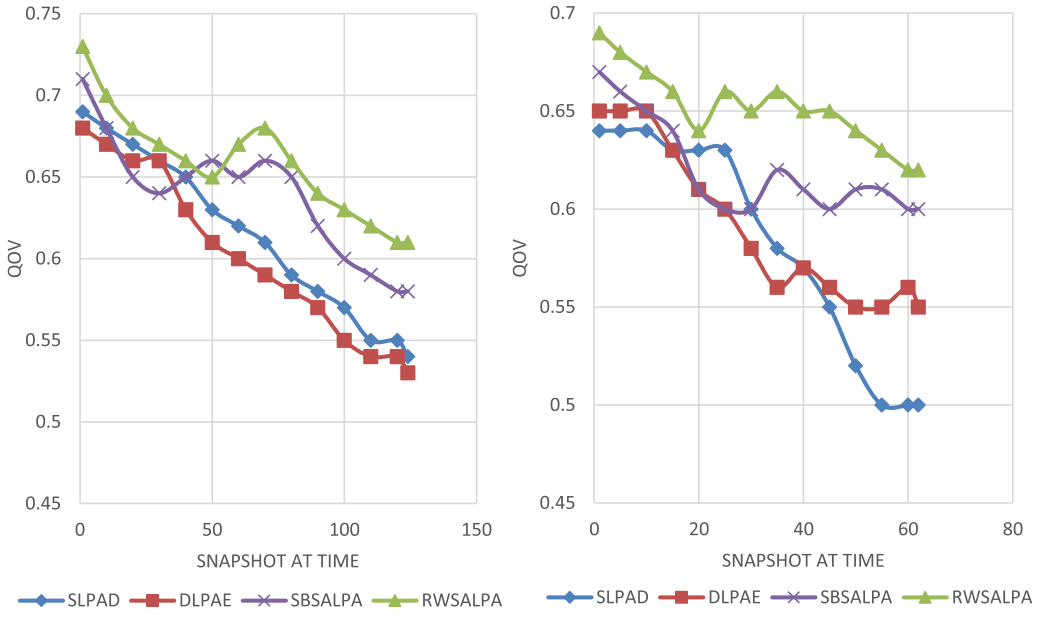
where symbols k_i and k_j are degree of vertexes *i* and *j* respectively. Symbol *m* is the number of edges and A_{ij} is the adjacency matrix. Symbols β_{ic} and β_{jc} express the strength of belonging nodes *i* and *j* to community *c* respectively. Symbols k_{ic} and k_{jc} are the total weight of links from nodes *i* and *j* to community *c* respectively.

To evaluate synthetic networks and DBLP network accuracy, the Normalized Mutual Information (NMI) measure is employed. This measure compares detected communities with ground-truth communities and computes their similarity. The higher the similarity between these communities, the higher the NMI value. When ground-truth communities of the network exist, the NMI is the most robust and accurate parameter to measure accuracy. In this paper, we aim to discover overlapping communities; therefore we use the overlapping extension of this measure which is defined as [39]:

$$NMI(X, Y) = 1 - \frac{H(XY) + H(YX)}{2} \tag{6}$$

where *X* and *Y* are random variables. *H(X|Y)* is the normalized conditional entropy of a cover *X* with respect to Y, and *H(Y|X)* is the normalized conditional entropy of a cover *Y* with respect to *X*. Symbol *H(X|Y)* is defined as [39]:

$$H(X \mid Y) = \frac{1}{|C'|} \sum_{K} \frac{H(X_k \mid Y)}{H(X_k)} \tag{7}$$



(a) Arxiv HEP-PH dataset (b) Arxiv HEP-TH dataset

Fig. 5. Modularity values achieved by various algorithms on Arxiv HEP-PH and Arxiv HEP-TH networks; RWSALPA, SBSALPA, DLPAE, SLPAD. (a) Arxiv HEP-PH dataset (b) Arxiv HEP-TH dataset.

where

$$H(X_k Y)=min_{l\in\{1,2,\ldots,|C''|}H(X_k Y_l) \tag{8}$$

Symbol $H(X_k \mid Y_l)$ is the conditional entropy of a cluster X_k given Y_l and symbol $H(X_k|Y)$ is the conditional entropy of a cluster X_k given Y. C' and C" are clusters. Symbol $H(Y|X)$ is defined in the same way. F1-score is another measure used in order to examine accuracy more precisely than NMI. The NMI Focus on the overall measure while the F1-score take attention to the node level. The range of F1-score values is between 0 and 1. F1(C_1, C_2) is the harmonic mean of precision and recall between cover-sets C_1, C_2. Cover-set C_1 is related to grand-truth covers and C_2 is related to the evaluated covers. We need to determine $C_i \in C_1$ corresponds to $C_j \in C_2$. We define $F1_{AVG}(C_1, C_2)$ to be the average of F1(C_1, C_2) of the best-matching ground-truth community to each detected community, and the F1-score of the best-matching detected community to each ground-truth community [40]:

$$precision(C_i, C_j) = \frac{|C_i \cap C_j|}{|C_i|} C_i \in C_1, \quad C_j \in C_2 \tag{9}$$

$$recall(C_i, C_j) = \frac{|C_i \cap C_j|}{|C_j|} C_i \in C_1, \quad C_j \in C_2 \tag{10}$$

$$F1(C_i, C_j) = \frac{2 . precision(C_i, C_j) . recall(C_i, C_j)}{recall(C_i, C_j) + precision(C_i, C_j)} C_i \in C_1, \quad C_j \in C_2 \tag{11}$$

$$F1_{AVG}(C_1, C_2) = \frac{1}{2 \mid C_1 \mid} \sum_{C_i \in C_1} maxiF1(C_i, C_2) + \frac{1}{2 \mid C_2 \mid} \sum_{C_j \in C_2} maxjF1(C_j, C_1) \tag{12}$$

where Symbols C_1 and C_2 determine evaluated and ground-truth cover node-sets respectively. Measure $precision(C_1, C_2)$ is the number of correctly detected overlapping nodes divided by the total number of detected overlapping nodes and $recall(C_1, C_2)$ is the number of correctly detected overlapping nodes divided by the true number of detected overlapping nodes.

5.3. Results

The results of implementing RWSALPA, SBSALPA, SLPAD [8] and DLPAE [9] on all available datasets including real and synthetic networks are shown in Figs. 5, 6, 7, 8, 9 and 10 respectively. We choose SLPAD and DLPAE because both of them are able to detect overlapping communities in dynamic social networks.

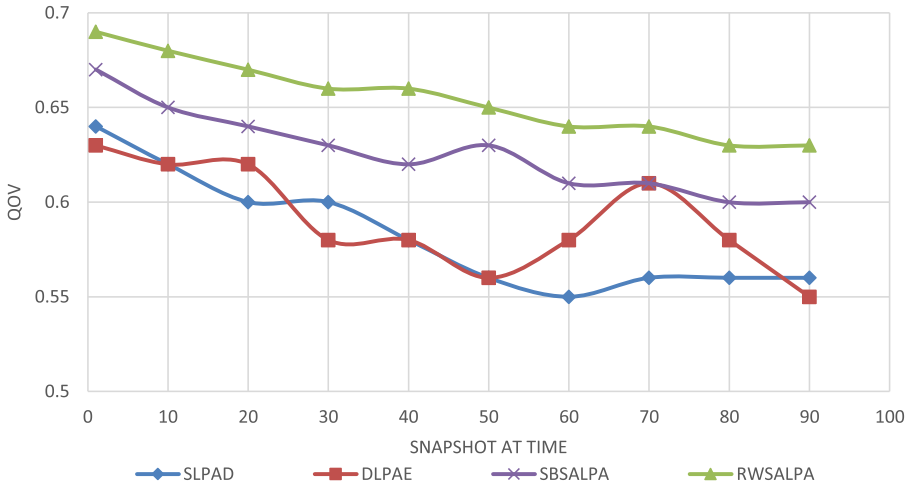


Fig. 6. Modularity values achieved by various algorithms on Enron network; RWSALPA, SBSALPA, DLPAE, SLPAD.

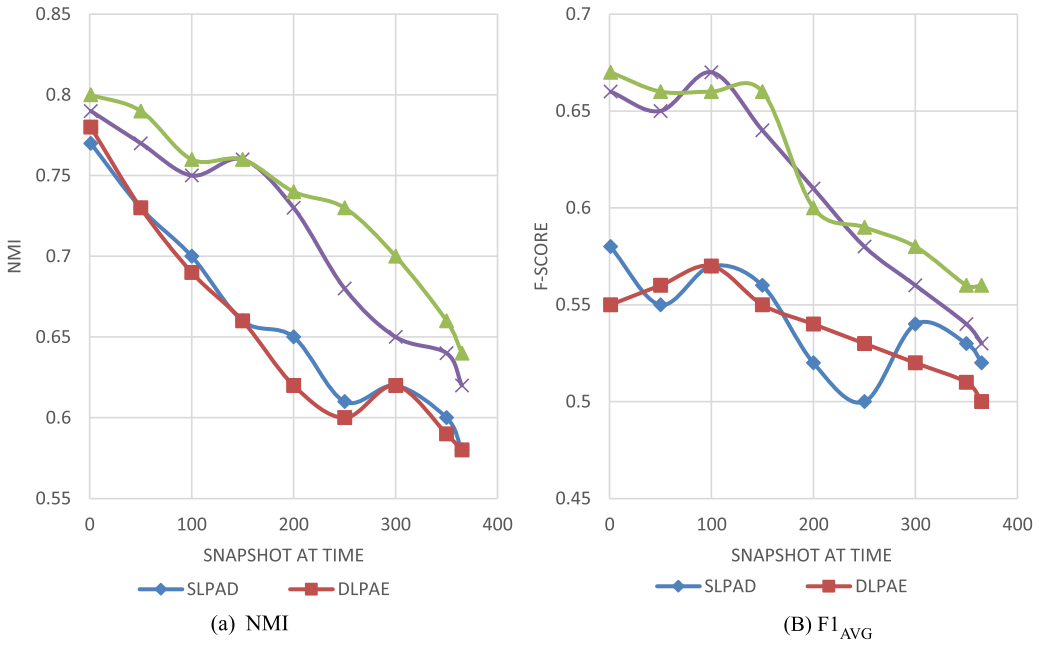
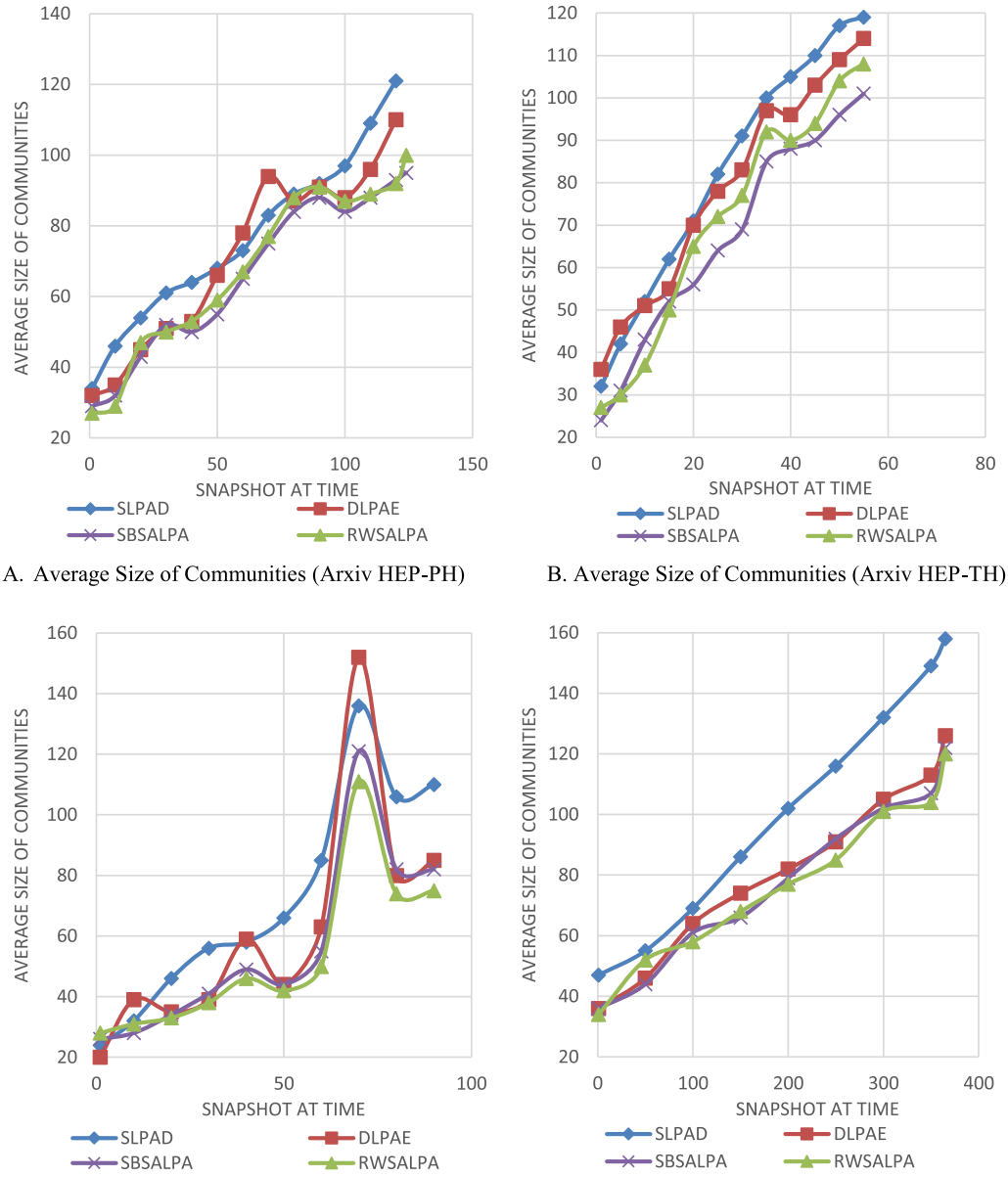


Fig. 7. The results achieved by applying various algorithms on DBLP network; RWSALPA, SBSALPA, DLPAE, SLPAD. (a) NMI (B) $F1_{AVG}$.

5.3.1. Results on real networks

To assess the performance of the proposed method, this study conducts experiments on four real networks.

5.3.1.1. Arxiv HEP-PH. The comparison of the modularity of two variations of the proposed method with other available methods on Arxiv HEP-PH dataset is plotted in Fig. 5(a). From the data in Fig. 5(a), it is apparent that all methods (the proposed method and other available methods) reveal a decreasing trend. That is why they in the first snapshot consider the whole graph while in snapshots 2 and onwards, consider the changed edges and nodes in comparison with the immediately previous snapshot. Therefore, as we takes changes instead of the whole graph in snapshots 2 and onwards, the modularity decreasing is inevitable. Moreover, the amount of modularity decreasing in RWSALPA and SBSALPA is more than in SLPAD [8] and DLPAE [9] in the earlier snapshots, such that the accuracy of SLPAD reaches close to that of RWSALPA and is greater than SBSALPA in snapshots 3 and 4. However, in the middle and last snapshots, the amount of modularity decreasing in two variations of the proposed method is less than other available methods. From above facts, it can be concluded that two variations of the proposed method achieve better performance in view of the node and edge changes than other available methods.



A. Average Size of Communities (Arxiv HEP-PH) B. Average Size of Communities (Arxiv HEP-TH)

C. Average Size of Communities (Enron Email) D. Average Size of Communities (DBLP)

Fig. 8. The average size of communities achieved by various algorithms on real networks; RWSALPA, SBSALPA, DLPAE, SLPAD. A Average Size of Communities (Arxiv HEP-PH) B. Average Size of Communities (Arxiv HEP-TH) C Average Size of Communities (Enron Email) D. Average Size of Communities (DBLP).

5.3.1.2. Arxiv HEP-TH. The comparison of the modularity of two variations of the proposed method with other available methods on Arxiv HEP-TH dataset is plotted in Fig. 5(b). In this dataset, the number of snapshots is half that of the Arxiv HEP-PH dataset while the number of nodes in the datasets is approximately same. Since in this dataset nodes are only added (not deleted), the number of new nodes is much lower than in the Arxiv HEP-PH dataset. In the earlier snapshots, SLPAD and DLPAE indicate an increasing trend while in the later snapshots, their accuracy decreased considerably. All variations of the proposed method achieve higher modularity in most snapshots except for the earlier snapshots. As in Arxiv HEP-PH, the number of snapshots is the twice of Arxiv HEP-TH and the difference between RWSALPA and SBSALPA in Arxiv HEP-PH is minor than Arxiv HEP-TH, it can be concluded that the more snapshots, the more similarity between the performance of RWSALPA and SBSALPA.

5.3.1.3. Enron Email. Enron Email is denser than other available datasets. Fig. 6 shows the experiment results on Enron Email networks. From the data in Fig. 6, it is apparent that all variations of the proposed method achieve higher modularity than other

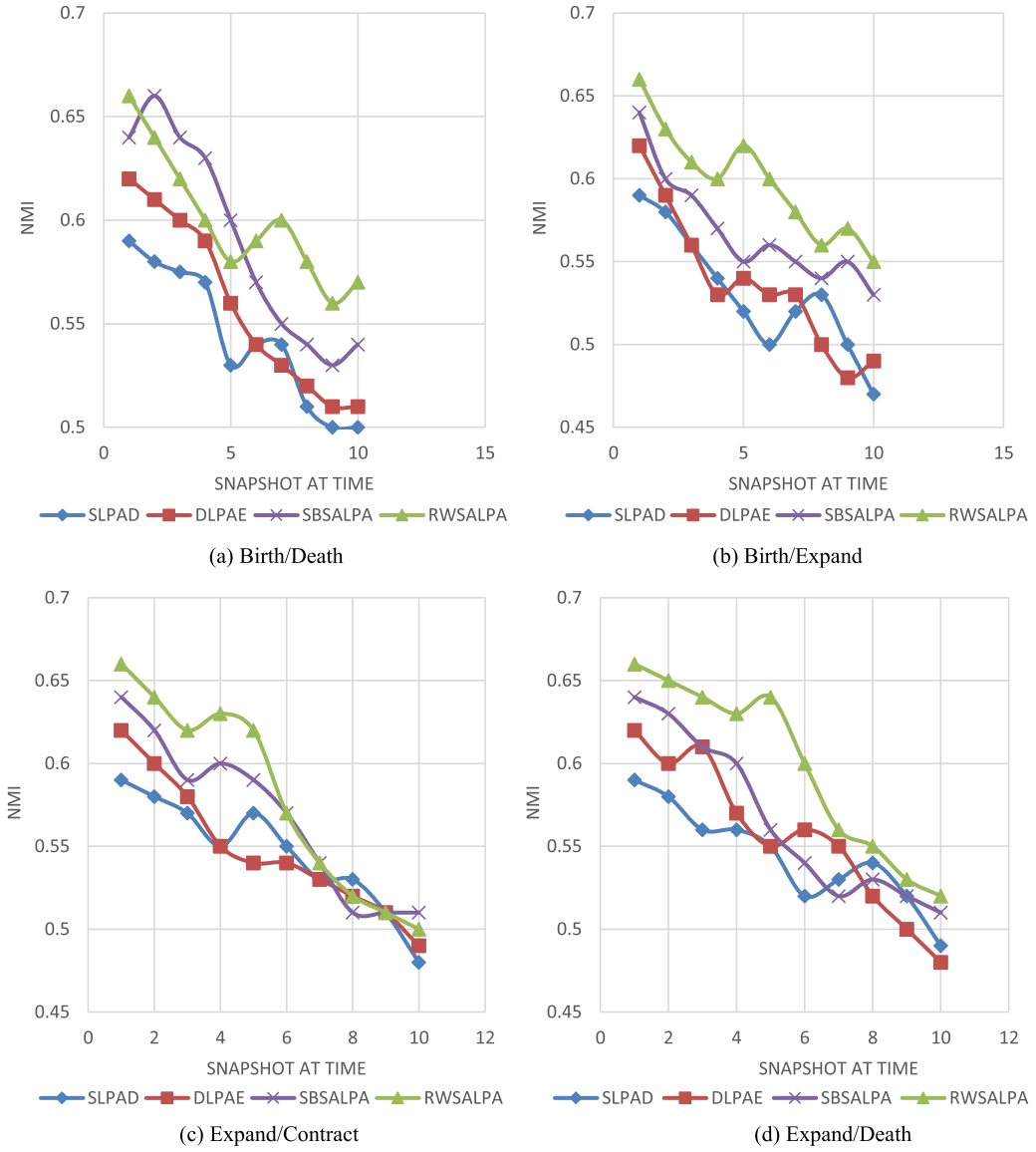


Fig. 9. NMI values achieved by various algorithms on Synthetic networks; RWSALPA, SBSALPA, DLPAE, SLPAD. (a) Birth/Death (b) Birth/Expand (c) Expand/Contract (d) Expand/Death.

available methods in all snapshots. This improvement demonstrates that RWSALPA and SBSALPA discover more accurate communities than SLPAD [8] and DLPAE [9] in highly overlapped communities. Moreover, in Enron Email, unlike citation networks, nodes can be added or deleted. Consequently, it can be concluded that deleting nodes produces no negative effect on all variations of the proposed method.

Based on above facts, in above three small networks, RWSALPA achieves better performance than the SBSALPA based on the modularity measure. However, Fig. 8 shows that in citation datasets, SBSALPA detects communities with lower size than RWSALPA. Therefore, in these datasets, RWSALPA detect more modular and larger communities than SBSALPA.

5.3.1.4. DBLP. Unlike other available real datasets, the DBLP dataset contains ground-truth communities; therefore we can apply two important measures: NMI and $F1_{AVG}$. The first measure considers the similarity between communities, and the latter is more precise and take nodes into account. Moreover, in this dataset, the number of nodes and snapshots is much greater than other available datasets; therefore, the author can evaluate the accuracy of all variations of the proposed method in a large-scale network. The results of applying the proposed method and other available methods in this dataset are plotted in Fig. 7. The results show that the NMI and $F1_{AVG}$ of RWSALPA are greater than those of other methods in most snapshots. From the above facts, it can be

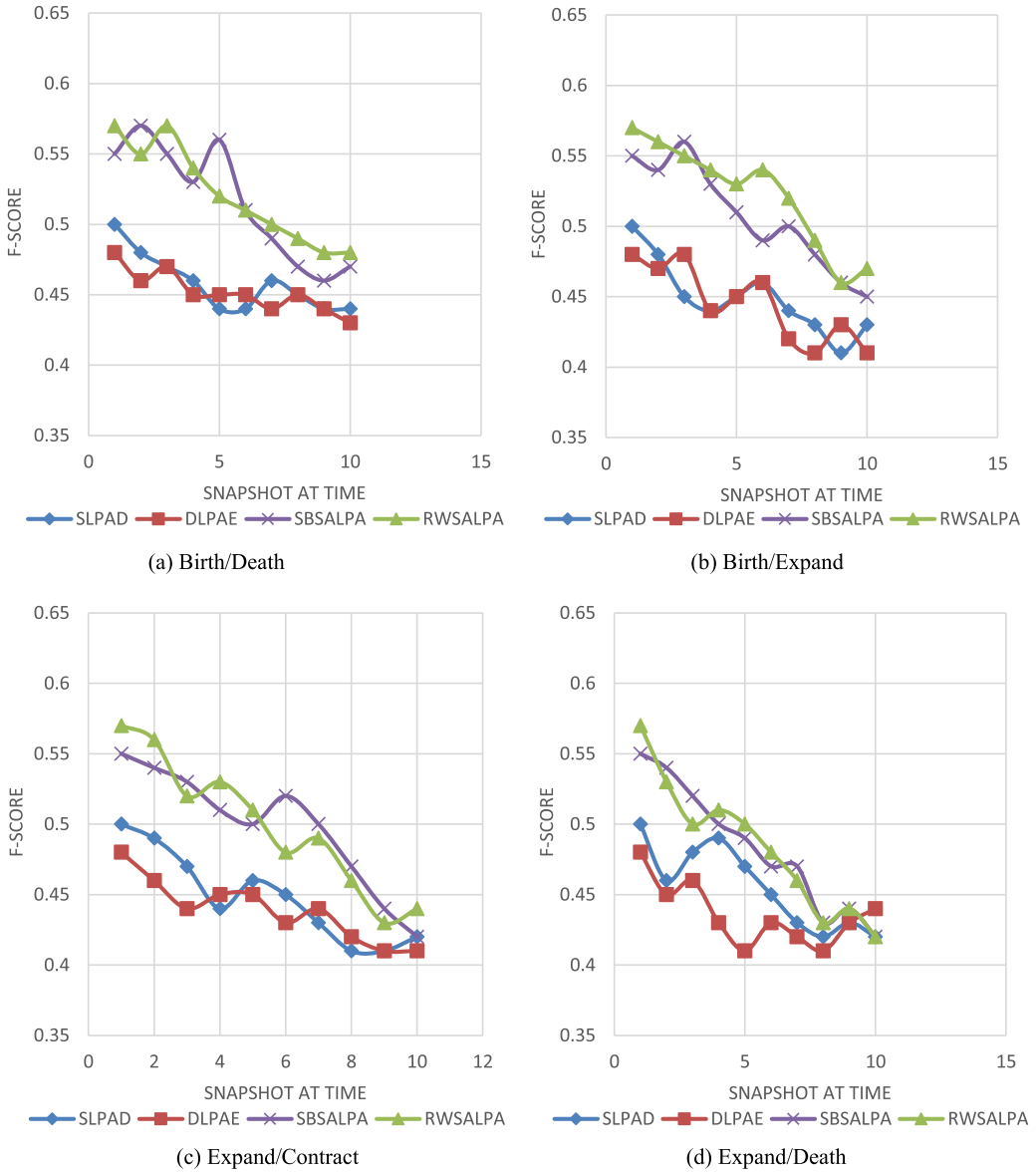


Fig. 10. $F1_{AVG}$ values achieved by various algorithms on Synthetic networks; RWSALPA, SBSALPA, DLPAE, SLPAD. (a) Birth/Death (b) Birth/Expand (c) Expand/Contract (d) Expand/Death.

Table 6.
Average running time (sec) achieved by various algorithms on real networks; SLPAD;DLPAE; SBSALPA;RWSALPA.

Methods Datasets	SLPAD	DLPAE	SBSALPA	RWSALPA
Arxiv HEP-PH	166	180	6189	6638
Arxiv HEP-TH	143	156	5763	5892
Enron Email	242	229	7912	8124
DBLP	550	581	35434	35736

concluded that from the nodes and communities perspective, two variations of the proposed method achieve better results than the two other methods in large-scale networks. That is why these variations of the proposed method consider the role of new nodes in order to spread their information. The RWSALPA achieves better performance than SBSALPA in the DBLP dataset. As it is a large-scale dataset, it can be concluded that RWSALPA is better candidate than SBSALPA for community detection in these networks.

The comparison of the average size of communities of two variations of the proposed method with other available methods on real networks is plotted in Fig. 8. Fig. 8 shows that RWSALPA and SBSALPA produce smaller communities than other available methods in real networks. This result represents that SLPAD and DLPAE tend more to forming large communities than RWSALPA and SBSALPA.

The comparison of the average running time of two variations of the proposed method with other available methods on real networks is plotted in Table 6. Table 6 shows that the variations of the proposed method detects community slower than other available methods in all real networks. For example, in DBLP dataset, the running time of RWSALPA and SBSALPA is more than fifty times slower than other available methods. That is because the third part of the proposed method traverses the graph for all nodes by breadth-first search.

5.3.2. Results on synthetic networks

In real networks, it is demonstrated that all variations of the proposed method improve the accuracy of community detection based on two measures: NMI and $F1_{AVG}$. As grand-truth communities in the majority of real networks is not available, we generate four synthetic dynamic networks. These networks are smaller than the DBLP network; therefore we are able to evaluate all variations of the proposed method on small networks with grand-truth communities.

These networks have ten snapshots, where 20 events occur in each snapshot. For evaluating different events effect on the variations of the proposed method and other available methods, we consider different event pairs in each network. In the first network, birth and death events occur while in the second network, birth and expand events occur. In the third network, expand and contract events and in the fourth network, expand and death are occurred.

The NMI is computed for each snapshot based on results of available methods on four dynamic synthetic networks (Fig. 9). The NMI of all variations of the proposed method is greater than that of these variations in the third and fourth networks. These results show the birth event has the positive effect on the proposed method. Moreover, From Fig. 8, it can be concluded that all variations of the proposed method achieve higher NMI than other available methods. Therefore, it shows that different events have no negative results on the NMI of all variations of the proposed method.

The $F1_{AVG}$ is computed for each snapshot based on results of available methods on four dynamic synthetic networks (Fig. 10). From Fig. 10, it can be concluded that all variations of the proposed method achieve higher $F1_{AVG}$ than other available methods. Therefore, it shows that different events have no negative results on the $F1_{AVG}$ of all variations of the proposed method.

In all networks except for Birth/Death network, RWSALPA achieves higher NMI than SBSALPA in the most snapshots. That is why in Birth/Death network, nodes tend more to forming communities than expanding them.

6. Conclusion

In this paper, we proposed the new method (SALPA) based on LPA and spreading activation to improve the accuracy of community detection in dynamic social networks. The proposed method had two variations, where each variation corresponds to a weighting algorithm. Using the proposed method, the new nodes had more chance than old nodes to spreading their labels to other nodes in social networks. In this method, each label had an activation value representing its strength of spreading. The results of the experiments indicated that the variations of the proposed method detect communities more accurate than other available methods while they are slower than other available methods. From among these variations, RWSALPA achieves better performance in comparison with SBSALPA in real networks based on the modularity measure while in real small networks, SBSALPA detects smaller communities than RWSALPA. Moreover, in synthetic networks, except for Birth/Death networks, RWSALPA detects more accurate communities than SBSALPA. From above facts, it can be concluded that RWSALPA is better candidate for community detection in comparison with SBSALPA and other available methods in most of the experimented networks.

A large number of community detection methods such as the proposed method ignores the direction of the edges in the graph. As in the second part of the proposed method, we used two weighting algorithms, the proposed method can applied on unweighted networks. Therefore, if we try to use weighted networks, we would ignore the second part of the proposed method. Detecting communities in dynamic networks is very challenging and discovering dynamic communities is still in its infancy and can be addressed by using methods such as the proposed method in the future.

Funding

The research was supported by University of Isfahan.

References

[1] B. Krishnamurthy, J. Wang, On network-aware clustering of web clients, ACM SIGCOMM Computer Communication Review 30 (4) (2000) 97–110.
[2] M.E. Newman, M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E 69 (2) (2004) 026113.
[3] U.N. Raghavan, R. Albert, S. Kumara, Near linear time algorithm to detect community structures in large-scale networks, Phys. Rev. E 76 (3) (2007) 036106.
[4] M.J. Barber, J.W. Clark, Detecting network communities by propagating labels under constraints, Phys. Rev. E 80 (2) (2009) 026129.
[5] I.X. Leung, P. Hui, P. Lio, J. Crowcroft, Towards real-time community detection in large networks, Phys. Rev. E 79 (6) (2009) 066107.
[6] S. Kelley, M. Goldberg, M. Magdon-Ismail, K. Mertsalov, A. Wallace, Defining and discovering communities in social networksIn Handbook of Optimization in Complex Networks (2012), 2012, pp. 139–168.

[7] J. Xie, Agent-based dynamics models for opinion spreading and community detection in large-scale social networks (Doctoral dissertation), Rensselaer Polytechnic Institute, 2012.
[8] N. Aston, J. Hertzler, W. Hu, Overlapping community detection in dynamic networks, J. Softw. Eng. Appl. 7 (10) (2014) 872.
[9] K. Liu, J. Huang, H. Sun, M. Wan, Y. Qi, H. Li, Label propagation based evolutionary clustering for detecting overlapping and non-overlapping communities in dynamic networks, Knowledge-Based Syst. 89 (2015) 487–496.
[10] A. Troussov, F. Dařena, J. Žižka, D. Parra, P. Brusilovsky, Vectorised Spreading Activation algorithm for centrality measurement, Acta Universitatis Agriculturae et Silviculturae Mendelianae Brunensis 59 (7) (2014) 469–476.
[11] S. Sun, J. Gong, J. He, S. Peng, A spreading activation algorithm of spatial big data retrieval based on the spatial ontology model, Cluster Comput. 18 (2) (2015) 563–575.
[12] P. De Meo, E. Ferrara, G. Fiumara, A. Provetti, Enhancing community detection using a network weighting strategy, Inform. Sci. 222 (2013) 648–668.
[13] N. Mozafari, A. Hamzeh, An enriched social behavioural information diffusion model in social networks, J. Inform. Sci. 41 (3) (2015) 273–283.
[14] R.I. Lung, C. Chira, A. Andreica, Game theory and extrenal optimization for community detection in complex dynamic networks, PloS One 9 (2014) 891–901.
[15] H. Alvari, A. Hajibagheri, G. Sukthankar, Community detection in dynamic social networks: a game-theoretic approach, In Adv. Soc. Netw. Anal. Mining (ASONAM), IEEE (2014) 101–107.
[16] N. Nisan, T. Roughgarden, E. Tardos, V.V. Vazirani, Algorithmic Game Theory 1, Cambridge University Press, *Cambridge*, 2007.
[17] R. Cazabet, F. Amblard, and C. Hanachi, Detection of overlapping communities in dynamical social networks, In International Conference on Social Computing (SocialCom), IEEE, pp. 309–314, 2010.
[18] H.S. Ma and J.W. Huang, CUT: community update and tracking in dynamic social networks, in: Proceedings of the 7th Workshop on Social Network Mining and Analysis, IEEE, p. 6, 2013.
[19] A. Clauset, M.E.J. Newman, C. Moore, Finding community structure in very large networks, Phys. Rev. E 70 (6) (2004) 1–6.
[20] N.P. Nguyen, T.N. Dinh, Y. Xuan, M.T. Thai, Adaptive algorithms for detecting community structure in dynamic social networks, In IEEE Conference on Computer Communications (INFOCOM), IEEE, pp. 2282–2290, 2011.
[21] Y.R. Lin, Y. Chi, S. Zhu, H. Sundaram, B.L. Tseng, Facetnet: a framework for analyzing communities and their evolutions in dynamic networks, in: Proceedings of the 17th International Conference on World Wide Web, ACM, pp. 685–694, 2008.
[22] A. Lancichinetti, F. Radicchi, J.J. Ramasco, S. Fortunato, Finding statistically significant communities in networks, PloS One 6 (4) (2011) e18961.
[23] M. Hosseini-Pozveh, K. Zamanifar, A.R. Naghsh-Nilchi, A community-based approach to identify the most influential nodes in social networks, J. Inform. Sci. 43 (2) (2016) 204–220.
[24] Y. Yi, Y. Shi, H. Zhang, J. Wang, J. Kong, Label propagation based semi-supervised non-negative matrix factorization for feature extraction, Neurocomputing 149 (2015) 1021–1037.
[25] S. Gregory, Finding overlapping communities in networks by label propagation, New J. Phys. 12 (10) (2010) 103018.
[26] J. Xie, B.K. Szymanski, X. Liu, SLPA: Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process, in: Proceedings of the 11th International Conference on Data Mining Workshops, IEEE, pp. 344–349, 2011.
[27] K. Kuzmin, S.Y. Shah, B.K. Szymanski, Parallel overlapping community detection with SLPA, In International Conference on Social Computing (SocialCom), IEEE, pp. 204–212, 2013.
[28] J. Xie, M. Chen, B.K. Szymanski, LabelRankT: Incremental community detection in dynamic networks via label propagation, In Proceedings of the Workshop on Dynamic Networks Management and Mining, ACM, pp. 25–32, 2013.
[29] J. Xie, B.K. Szymanski, LabelRank: a stabilized label propagation algorithm for community detection in networks (In Network Science Workshop (NSW)IEEE (2013) 138–143.
[30] S. He-Li, H. Jian-Bin, T. Yong-Qiang, S. Qin-Bao, L. Huai-Liang, Detecting overlapping communities in networks via dominant label propagation, Chin. Phys. B 24 (2015) 551–559.
[31] J.H. Holland, Building blocks, cohort genetic algorithms, and hyperplane-defined functions, Evol. Comput. 8 (4) (2000) 373–391.
[32] J. Gehrke, P. Ginsparg, J. Kleinberg, Overview of the 2003 KDD Cup, ACM SIGKDD Explorations Newsletter 5 (2) (2003) 149–151.
[33] J. Leskovec, J. Kleinberg and C. Faloutsos, Graphs over Time: Densification Laws, Shrinking Diameters and Possible Explanations, ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), ACM, pp. 177–187, 2005.
[34] B. Klimmt, Y. Yang, The enron corpus: a new dataset for email classification research (In Proceedings of the European Conference on Machine Learning (ECML), Springer Berlin Heidelberg, 2004, pp. 217–226.
[35] J. Yang, J. Leskovec, Defining and evaluating network communities based on ground-truth, Knowledge Inform. Syst. 42 (1) (2015) 181–213.
[36] A. Lancichinetti, S. Fortunato, Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities, Phys. Rev. E 80 (1) (2009) 016118.
[37] M. Takaffoli, J. Fagnan, F. Sangi, O.R. Zaïane, Tracking changes in dynamic information networks, In Computational Aspects of Social Networks (CASoN), IEEE (2011) 94–101.
[38] D. Chen, M. Shang, Z. Lv, Y. Fu, Detecting overlapping communities of weighted networks via a local algorithm, Phys. A: Stat. Mech. Appl. 389 (19) (2010) 4177–4187.
[39] A. Lancichinetti, S. Fortunato, J. Kertesz, Detecting the overlapping and hierarchical community structure in complex networks, New J. Phys. 11 (3) (2009) 033015.
[40] J. Yang and J. Leskovec, Community-affiliation graph model for overlapping network community detection," in International Conference on Data Mining (ICDM), IEEE, pp. 1170–1175, 2012.