Physics Letters A ••• (••••) •••–•••

Contents lists available at ScienceDirect

Physics Letters A

www.elsevier.com/locate/pla

Directed LPA: Propagating labels in directed networks

Xue Li

Computer Science, Northeast Forestry University, Harbin, 150040, China

ARTICLE INFO

Article history:
Received 5 July 2018
Received in revised form 29 November 2018
Accepted 29 November 2018
Available online xxxx
Communicated by C.R. Doering

Keywords:
Directed networks
Community detection
Directed label propagation
Directed modularity

ABSTRACT

Ignoring edge directionality and considering the graph as undirected is a common approach to detect communities in directed networks. However, it's not a meaningful way due to the loss of information captured by the edge property. Even if Leicht and Newman extended the original modularity to a directed version to address this issue, the problem of distinguishing the directionality of the edges still exists in maximizing modularity algorithms. To this direction, we extend one of the most famous scalable algorithms, namely label propagation algorithm (LPA), to a directed case, which can recognize the flow direction among nodes. To explore what properties the directed modularity should have, we also use another directed modularity, called LinkRank, and provide an empirical study. The experimental results on both real and synthetic networks demonstrate that the proposed directed extension algorithms can not only make use of the edge directionality but also keeps the same time complexity as LPA.

1. Introduction

Complex networks appear in the real world widely and its theory provides a new view to study the complex system. As the building blocks of complex networks, community structure has attracted much interest throughout the recent years. However, the research progress of the community detection in directed and undirected networks makes a big difference. The main focus of the current research is on undirected graphs. As Santo Fortunato stated that developing methods of community detection for directed graphs is a hard task [1]. A common approach is to ignore the direction of the link and run the algorithms designed for undirected networks, largely due to no other better options. Thus the potentially useful information of the edge directions is discarded and the meaningful communities are also missed.

To tackle the above problem, Leicht and Newman extended the original modularity [2] to a directed version [3] and defined as below:

$$DQ = \frac{1}{m}\sum_{ij}[A_{ij} - \frac{k_i^{in}k_j^{out}}{m}]\delta_{(c_i,c_j)} \tag{1}$$

where A is the adjacency matrix of the network, k_i^{in} and k_j^{out} are the in- and out-degree of the vertices.

Directed modularity (DQ) takes both the edges within communities and the edge direction into consideration. The crucial point is that an edge from a low out-degree but high in-degree node to an opposite case node should be considered a bigger surprise than vice versa. However, the above idea is not fully realized. Kim et al. [4] observed that DQ cannot properly discriminate the direction of the edges. They proposed a new directed modularity, called LinkRank modularity (LQ), which is similar to Google's PageRank algorithm [5]. The community is defined as a group of nodes where a random walk prefers to stay. The contribution of an edge (i, j) to community formation can be defined as:

$$L_{ij} = \pi_i G_{ij} \tag{2}$$

where π_i is the i-th element of PageRank vector, G_{ij} is the element of Google matrix.

The definition of this directed modularity can be abstracted as below:

$$\begin{aligned} LQ = &\textit{(fraction of time for walking within communities)} \\ &- \textit{(expected value of this fraction)} \end{aligned} \tag{3}$$

Therefore, the directed modularity can be expressed as:

$$LQ = \sum_{ij}[L_{ij} - \pi_i\pi_j]\delta_{(c_i,c_j)} \tag{4}$$

where $\pi_i\pi_j$ is the expected probability for a walk moving from i to j.

Although DQ is more highly cited, LQ is a pretty stricter criterion. They will be both used to evaluate the identified communities and make a comparison.

E-mail address: nefu_education@126.com.

https://doi.org/10.1016/j.physleta.2018.11.047
0375-9601/

This paper tries to explore a heuristic way for LPA to make good use of the directionality. Considering the direction of arcs when detecting community is meaningful, and as far as I know, very few works were dedicated to this topic. By transforming the edge direction into an edge weight, the modified LPA can identify the local flow direction. For a linear complexity, we don't do many changes to LPA, just resetting the label choosing rule. To demonstrate the extensibility of directed LPA, we propose a constrained directed label propagation algorithm (CDLPA), which adopts a strategy to alleviate the over propagation problem of LPA. Furthermore, to explore what properties the directed modularity should have, we make a full comparison between the directed modularity (DQ) and LinkRank (LQ).

In the next section, the theory of the original label propagation and its drawbacks are discussed. Section 3 explains and justifies the design of our algorithms. In section 4, we present the results of experiments that show how the proposed algorithms behave and measure their performance, comparing these with other community detection algorithms. Our conclusions appear in section 5.

2. Label propagation algorithm

2.1. The original label propagation algorithm

Label propagation algorithm [6] is one of the fastest algorithms in community detection for undirected networks. It can compute communities for large-scale networks and be coded with a few lines on individual computers, which is true only for a few algorithms in the literature. LPA only considers the topology, requiring less extra information about the network. It works as follows:

Step1: Initializing each node with a unique label.
Step2: Every node chooses a label among its neighbors based on the frequency of occurrence.
Step3: If the distribution of the labels reach a steady state, then stop the algorithm, or back to Step2.

Finally, compute communities by the identical labels.

The label choosing principle is expressed as below:

$$l_v^{new} = \underset{l}{argmax} |N^l(v)| \tag{5}$$

where v is a node, l denotes the label of a node, N_v is the neighbors of v.

2.2. Over propagation and giant community

One of the most obvious drawbacks of LPA is the over propagation problem. The main reason behind the over propagation is the rapid and aggressive expansion of the core of some communities. The less extensible cores or weaker communities have little chance to grow and even are swallowed by other communities. The extreme case of the over propagation is one giant community, dividing all the nodes into one class. However, comparing with other cases, the one giant community is not so bad. It at least tells you the community detection failed in this attempt and you need another try.

The over propagation phenomenon exists in both undirected and directed community detections. To alleviate this problem, in the following sections, we adopt a strategy in the paper [7], which designs a growth capacity for communities, starting from a small capacity and increasing it over iterations. The over propagation phenomenon exists in both undirected and directed community detection. To alleviate this problem, in the following sections, we adopt a strategy in the paper [7], which designs a growth capacity for communities, starting from a small capacity and increasing it over iterations.

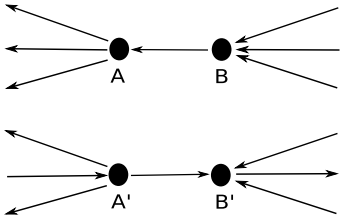

Fig. 1. Nodes A, B, A', and B' are four nodes in a directed binary network. The out- and in-degree of nodes are $i_A^{out} = i_{A'}^{out} = i_B^{in} = i_{B'}^{in} = 3$ and $i_A^{in} = i_{A'}^{in} = i_B^{out} = i_{B'}^{out} = 1$.

3. Directed extensions of label propagation algorithm

Based on the above observations, a heuristic algorithm, called Directed LPA, is proposed. To demonstrate its extensibility, we put forward a constrained Directed LPA.

3.1. Directed label propagation algorithm (DLPA)

As mentioned earlier, the inspiration of DLPA comes from the thoughts of directed modularity (DQ). In Fig. 1, according to the original idea of DQ, E_{BA} (edge B $\rightarrow$ A) should contribute more to modularity than $E_{A'B'}$ (edge $A' \rightarrow B'$). However, DQ may not work as expected. According to Eq. (1), we calculate the directed modularity of E_{BA} and $E_{A'B'}$ as below:

$$DQ_{BA} = \frac{1}{M}[0 - \frac{k_A^{in} k_B^{out}}{M}] + \frac{1}{M}[1 - \frac{k_B^{in} k_A^{out}}{M}] = \frac{1}{M}(1 - \frac{10}{M}) \tag{6}$$

$$DQ_{A'B'} = \frac{1}{M}[1 - \frac{k_{A'}^{in} k_{B'}^{out}}{M}] + \frac{1}{M}[\frac{k_{B'}^{in} k_{A'}^{out}}{M}] = \frac{1}{M}(1 - \frac{10}{M}) \tag{7}$$

It is obvious that DQ fails to distinguish the local flow direction and this may lead to not proper evaluation of the network structure. For the directed network community detection, it is necessary to take into account the direction of links. To this direction, we design a weight calculation rule for each edge:

$$1 - \frac{E_s^{out} * E_t^{in}}{k_s * k_t} \tag{8}$$

where E_s^{out} is the out-degree of the source node, E_t^{in} is the in-degree of the target node, k_s is the degree of source code, and k_t is the degree of target node.

We can also rewrite this strategy for each node to embed it in iterative computation. It can be written as below:

$$l_i^{new} = \underset{l}{argmax} \begin{cases} \sum_{j \in N^l_{(i^{in})}} [1 - \frac{k_i^{in} k_j^{out}}{k_i k_j}], & \textit{for in-link neighbors with label } l \\ \sum_{j \in N^l_{(i^{out})}} [1 - \frac{k_i^{out} k_j^{in}}{k_i k_j}], & \textit{for out-link neighbors with label } l \end{cases} \tag{9}$$

where k_i^{in} and k_i^{out} are in- and out-degree of node i, $N^l_{(i^{in})}$ and $N^l_{(i^{out})}$ are in- and out-link neighbors of node i with label l.

Then the contribution of E_{BA} and $E_{A'B'}$ to community formation can be represented as w_{BA} and $w_{A'B'}$ and computed as follows:

$$w_{BA} = 1 - \frac{k_B^{out} k_A^{in}}{k_B * k_A} = 1 - \frac{1 * 1}{4 * 4} = \frac{15}{16} \tag{10}$$

$$w_{A'B'} = 1 - \frac{k_{A'}^{out} k_{B'}^{in}}{k_{B'} * k_{A'}} = 1 - \frac{3 * 3}{4 * 4} = \frac{7}{16} \tag{11}$$

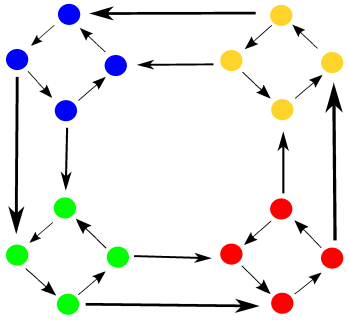

Fig. 2. We extract this simple directed graph from paper [8] and the weights are ignored.

It is clear that E_{BA} contributes more to identify local flow direction than $E_{A'B'}$.

The process of DLPA is the same as LPA except for label choosing rule. DLPA doesn't make too many changes to LPA and keeps the same time complexity.

3.2. Constrained directed label propagation algorithm (CDLPA)

We previously mentioned that LPA and its extensions suffer from over propagation phenomenon. The imbalance growth of communities is the key problem. To ensure the weaker cores of communities have a chance to grow, we adopt a strategy in paper [7] that sets a growth capacity for all communities in each iteration. The capacity function $C(t)$ is defined as below:

$$C(t) = ([\frac{kt}{T}] + 1) * \frac{N}{k} \tag{12}$$

where N is the number of nodes, t represents the t-th iteration, T is the maximal number of iterations, and k is the number of times we change the capacity of communities.

As the Eq. (12) shows that we increase the community capacity every $\frac{T}{k}$-th iteration by $\frac{N}{k}$. In each iteration, the algorithm will check whether any community reaches the current capacity limitations or not. If the size of a community reaches the current capacity, that community cannot attract new nodes until the capacity increases in the next iterations. This strategy tries to make a balance community growth. We set the community growth strategy for DLPA and propose an extension of DLPA, namely constrained directed LPA (CDLPA).

3.3. Case that makes a difference

To demonstrate the case that makes a difference, a simple directed network composed of sixteen nodes is introduced. As shown in Fig. 2, the manifold of the network is pretty clear. We run DLPA, CDLPA, and LPA on this toy network for hundreds of times.

$$LQ^{min}_{DLPA} = LQ^{max}_{DLPA} = 0.4299 \tag{13}$$

$$LQ^{min}_{CDLPA} = LQ^{max}_{CDLPA} = 0.4299 \tag{14}$$

$$LQ^{min}_{LPA} \approx 0.1, LQ^{max}_{LPA} \approx 0.3756 \sim 0.4299 \tag{15}$$

DLPA and CDLPA succeed in retrieving the communities as the color denoted whereas LPA generates dozens of results and in most cases, it fails to recognize the expected communities.

3.4. Time complexity

The difference between LPA, DLPA, and CDLPA is the label choosing principle. The complexity of the directed extensions of LPA is unchanged. The label initialization requires $O(n)$ time. Choosing a node randomly needs $O(1)$ time. In each iteration, edge weight or node similarity calculating based on the label choosing principle takes $O(m)$. Selecting a node label for a node requires $O(d)$, where d is the degree of a node. The number of iterations needed for the algorithms is equal to the total number of effective updates k. For CDLPA, the community growth strategy sometimes may delay the converge and the number of iteration is slightly bigger. So, loosely calculating, the total running time of these algorithms will be $O(k * max(n, m))$.

4. Experimental results

To fully demonstrate the differences between DLPA, CDLPA, and LPA, we run them on both real and synthetic networks.

4.1. Measures

In this paper, two kinds of directed modularity are mentioned. Although Kim et al. observed the drawbacks of DQ and proposed LQ, the differences between DQ and LQ are not exactly tested on the actual data. So they are both used to evaluate the identified community structure.

For efficiency of calculating, DQ in a partitioning with C clusters can be written as below:

$$DQ = \sum_{c_i \in C} [\frac{I_{c_i}}{M} - \frac{c_i^{in} c_i^{out}}{M^2}] \tag{16}$$

where I_{c_i} is number of links within partition c_i, c_i^{in} and c_i^{out} are in-degree and out-degree of partition c_i.

LQ is rewritten as below for the first time:

$$LQ = \sum_{t}^{C} [\sum_{i \in t} (\frac{\alpha \pi_i E_i}{i^{out}} + N_t \pi_i \frac{\alpha(0^{i^{out}}) + 1}{N}) - (\sum_{i \in t} \pi_i)^2] \tag{17}$$

where α is a damping factor and equals to 0.85, E_i is the number of out-links of node i within partition t, N_t is the number of nodes within partition t, π_i is i-th element of PageRank vector of the network.

In the synthetic network test part, the normalized mutual information (*NMI*) [9] is also used to evaluate the results of algorithms. Consider x and y as two partitions, then *NMI* can be defined as the fraction of the mutual information I and the conditional entropy H:

$$NMI(x, y) = \frac{I(x, y)}{\sqrt{H(x)H(y)}} \tag{18}$$

4.2. Tests on real-world networks

We have tested directed extension algorithms against LPA on a wide variety of networks (networks are download from http://konect.uni-koblenz.de) which are involved in different fields such as traffic, biology, paper citation, and social networks. Table 1 lists the details of these networks. Table 2 shows the evaluation for the identified community structures of these networks.

To have a better understanding of outputs from the algorithms, the maximum, average, and variance of modularity are computed for the ten datasets averaged on 100 or 50 realizations based on the size of networks.

In Table 2, the performances of DLPA and CDLPA are similar. For directed modularity (DQ), they outperform LPA in Max_DQ or Ave_DQ for about half of the networks. The directionality of the DLPA and CDLPA is not obvious. In Cora and Hepth networks, LPA performs better in both Max_DQ and Ave_DQ.

Table 1
Overview of the networks used in the experiments.

Network	Description	Nodes	Edges	$\overline{k^{in}}$
Air Traffic	USA air traffic control information	1225	2604	2
Bio-yeast	Protein–protein interactions	1458	1948	2
Human-protein	Protein interactions in Humans	2238	6425	3
Open Flights	Flight records	2398	30499	10
Twitter lists	Twitter user following information	22370	33101	2
Cora	Cora citations	23166	91500	4
Google	Google+ user–user links	23628	39242	2
Linux	Network of Linux source code files	30837	213954	7
Hepth	High Energy Physics Archive	34546	421578	12
Gnutella	Network of Gnutella hosts	62586	147892	3

For LinkRank modularity (LQ), as stressed in the table, directed algorithms performs better than LPA in almost all the items. The average LQ of DLPA and CDLPA is greater than the largest LQ of LPA for all tests. The directionality of these directed algorithms can be observed.

There are some intuitive differences between DQ and LQ. The evaluation of DQ for DLPA and CDLPA is not very positive. During the computation, we find the best partitions recognized by DQ and LQ sometimes are different for the same network, which is consistent with the observation of Kim et al. [4], that DQ cannot distinguish the direction of links. LQ can properly recognize the directionality of algorithms and the stability of LQ evaluation is much better. However, on the numerical level of community structural strength evaluation, DQ performs better. For the networks of Twitter and Human protein, the LQ is less than 0.1 and even less than 0.01 for the Google network. These structures seem so weak that they are not worth exploring at all. So as the criteria of directed community structures, directionality, and numerical evaluation level should be better taken into consideration.

4.3. Tests on synthetic networks

The synthetic networks employed in this part are generated by directed LFR benchmark [10]. The parameters of the LFR model including number of nodes (N), average in-degree ($< k^{in} >$), maximum in-degree (k^{in}_{max}), mixing factor (μ), exponent for the degree sequence ($t1$), exponent for the community size distribution ($t2$), and community sizes (c_{min}, c_{max}). We generate 6 directed networks with the following parameters:

$$N = 1000, < k^{in} >= 5, k^{in}_{max} = 10, t1 = 2, t2 = 1,$$
$$c_{min} = 10, c_{max} = 50,\ \mu \in [0.1\text{–}0.5]$$
$$N = 1000, < k^{in} >= 5, k^{in}_{max} = 10, t1 = 2, t2 = 1,$$
$$c_{min} = 20, c_{max} = 100,\ \mu \in [0.1\text{–}0.5]$$
$$N = 1000, < k^{in} >= 5, k^{in}_{max} = 50, t1 = 2, t2 = 1,$$
$$c_{min} = 10, c_{max} = 50,\ \mu \in [0.1\text{–}0.5]$$
$$N = 1000, < k^{in} >= 5, k^{in}_{max} = 50, t1 = 2, t2 = 1,$$
$$c_{min} = 20, c_{max} = 100,\ \mu \in [0.1\text{–}0.5]$$
$$N = 5000, < k^{in} >= 5, k^{in}_{max} = 50, t1 = 2, t2 = 1,$$
$$c_{min} = 10, c_{max} = 50,\ \mu \in [0.1\text{–}0.5]$$
$$N = 5000, < k^{in} >= 5, k^{in}_{max} = 50, t1 = 2, t2 = 1,$$
$$c_{min} = 20, c_{max} = 100,\ \mu \in [0.1\text{–}0.5]$$

The mixing factor μ can significantly affect the properties of the network and the larger it is, the less clear the community structure is.

As shown in Table 3, for the sake of completeness, we also compare with Louvain [11], Directed Louvain (D_Louvain) [12],

Table 2
Comparison between DLPA and LPA based on DQ and LQ.

Datasets	Methods	Max_DQ	Ave_DQ	Var_DQ	Max_LQ	Ave_LQ	Var_LQ
Air Traffic	LPA	0.4754	0.4383	0.0002	0.3578	0.3329	0
	DLPA	**0.5255**	**0.4860**	0.0002	**0.4261**	**0.4027**	0
	CDLPA	**0.5303**	**0.4862**	0.0002	**0.4244**	**0.4021**	0
Bio-yeast	LPA	0.6638	0.6371	0	0.3464	0.3289	0
	DLPA	**0.7144**	**0.6987**	0	**0.4200**	**0.4129**	0
	CDLPA	**0.7091**	**0.6977**	0	**0.4188**	**0.4131**	0
Human-protein	LPA	0.4189	0.1874	0.0068	0.0699	0.0417	0.0001
	DLPA	0.3540	**0.3391**	0	**0.0824**	**0.0800**	0
	CDLPA	0.3425	**0.3331**	0	**0.0818**	**0.0800**	0
Open Flights	LPA	0.6099	0.5353	0.0054	0.5793	0.5346	0.0021
	DLPA	0.5975	**0.5571**	0.0046	0.5770	**0.5502**	0.0021
	CDLPA	**0.6156**	0.5259	0.0086	**0.5868**	0.5315	0.0035
Twitter lists	LPA	0.7747	0.7567	0	0.0274	0.0268	0
	DLPA	**0.8351**	**0.8258**	0	**0.0296**	**0.0293**	0
	CDLPA	**0.8356**	**0.8257**	0	**0.0296**	**0.0293**	0
Cora	LPA	0.6565	0.6433	0	0.4668	0.4466	0.0001
	DLPA	0.6046	0.5878	0	**0.5038**	**0.4959**	0
	CDLPA	0.6042	0.5913	0.0001	**0.5025**	**0.4955**	0
Google	LPA	0.6435	0.6204	0.0003	0.0033	0.0032	0
	DLPA	**0.6748**	0.5737	0.0122	**0.0038**	**0.0034**	0
	CDLPA	**0.667**	0.5723	0.0084	**0.0037**	**0.0034**	0
Linux	LPA	0.1462	0.1163	0.0003	0.0882	0.0750	0
	DLPA	**0.2698**	**0.2129**	0.0015	**0.3561**	**0.2066**	0.0113
	CDLPA	**0.1897**	**0.1810**	0	**0.1275**	**0.122**	0
Hepth	LPA	0.6707	0.6551	0	0.4566	0.4474	0
	DLPA	0.6667	0.6546	0	**0.4651**	**0.4568**	0
	CDLPA	0.6629	0.6466	0.0001	**0.4603**	**0.4556**	0
Gnutella	LPA	0.3529	0.3263	0.0055	0.0928	0.0856	0.0004
	DLPA	**0.3859**	**0.3850**	0	**0.1174**	**0.1172**	0
	CDLPA	**0.3860**	**0.3855**	0	**0.1174**	**0.1173**	0

Table 3
Overview of the algorithms used in the experiments.

Algorithm	Property	Complexity
Infomap	Directed edges: True; Weighted edges: True	O(n*(n + m))
Louvain	Directed edges: False; Weighted edges: True	O(n*log n)
D_Louvain	Directed edges: True; Weighted edges: True	O(n*log n)
LPA	Directed edges: False; Weighted edges: True	O(max(n, m))
DLPA	Directed edges: True; Weighted edges: True	O(max(n, m))
CDLPA	Directed edges: True; Weighted edges: True	O(max(n, m))

and Infomap [8]. Louvain is an undirected modularity optimization algorithm. Directed Louvain is a directed extension of Louvain, maximizing DQ. Infomap is a state-of-the-art algorithm in directed community detection. For LPA, Louvain, and Infomap, we use the packages in python-igraph. DLPA and CDLPA are programmed in python, and D_Louvain is coded in C. The NMI is computed for these algorithms averaged on 50 or 2 realizations based on the complexity of algorithms and the size of networks.

From the point of view of data fluctuation, the fluctuations of average NMI curves in the left column images are smoother. The performances of these algorithms in the right column images decrease at different degrees with the size of community increases. As shown in Fig. 3(a), (c), (e), with the increase of max in-degree or number of nodes, the performances of DLPA and CDLPA get better and better. They can outperform LPA, Louvain, D_Louvain, and Infomap. However, when increasing the number of communities, as shown in Fig. 3(b), (d), (f), the network structure becomes more complex and the performances of DLPA and CDLPA are on a decrease and make a difference. Among these algorithms, the original LPA and its directed extensions show signs of failure at about $\mu = 0.5$ obviously, meaning that the community structure is not so clear and some monster communities are formed. Compared with DLPA, CDLPA behaves better and the application of growth capacity improves the overall performance obviously.

In this paper, the proposed algorithms do not change the original LPA too much to keep the low complexity. As the Fig. 3 shows, in most cases, DLPA outperforms LPA slightly. Interestingly, similar results can also be observed in D_Louvain and Louvain for the NMI test. However, it surely proves that the pure direction recognized strategy can have positive contributions to community detection. When we add other simple strategies to DLPA, the extension of DLPA (CDLPA) behaves a little better but takes more iterations. So, for better results, we can further integrate it with other knowledge.

Amongst these algorithms, the performance of Infomap seems pretty satisfied, but its complexity is so high that it is not suitable for the large networks. As Fig. 3 shows, when the community structure is pretty clear, DLPA and CDLPA can be a substitute for Infomap. When the community structure is not so clear and the network is sparse, DLPA or D_Louvain is also a good choice.

5. Conclusion and future works

This paper explores the way for LPA to make good use of directionality in directed community detection. A heuristic algorithm called DLPA is proposed, which can recognize the local flow direction among the nodes. To demonstrate the extensibility of DLPA, we propose a constrained DLPA to overcome the problems existed in LPA and DLPA. In fact, besides the community growth strategy, we can also integrate many other strategies into DLPA such as

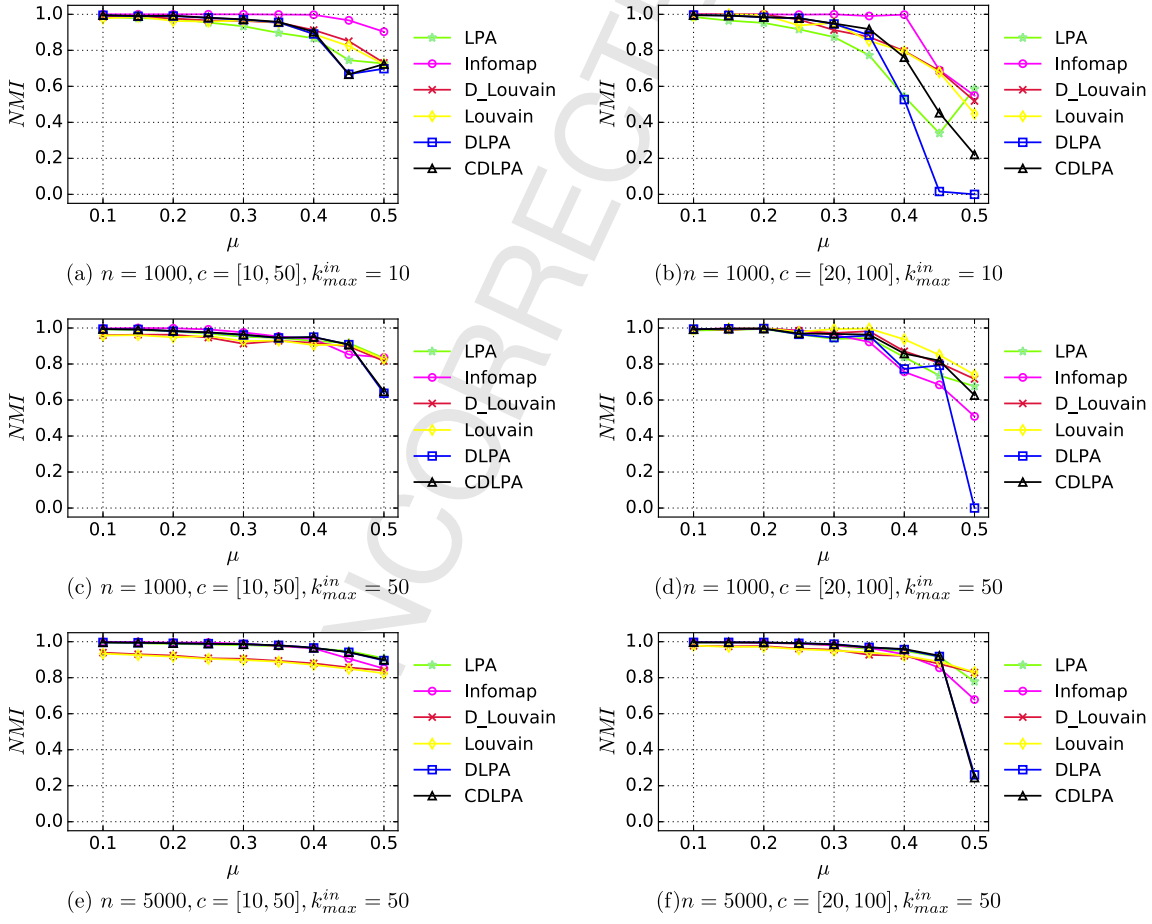

Fig. 3. Average NMI comparison on synthetic networks.

center nodes locating [13], a tunable weighting strategy [14], a potential game-based weighted Modularity optimization [15], and so on. The experiments on real and synthetic networks show that the proposed algorithms have a better performance than some of the current representative algorithms.

References

[1] S. Fortunato, Community detection in graphs, Phys. Rep. 486 (3–5) (2010) 75–174.
[2] M.E. Newman, Modularity and community structure in networks, Proc. Natl. Acad. Sci. 103 (23) (2006) 8577–8582.
[3] E.A. Leicht, M.E. Newman, Community structure in directed networks, Phys. Rev. Lett. 100 (11) (2008) 118703.
[4] Y. Kim, S.-W. Son, H. Jeong, Finding communities in directed networks, Phys. Rev. E 81 (1) (2010) 016103.
[5] A.N. Langville, C.D. Meyer, Google's PageRank and Beyond: The Science of Search Engine Rankings, Princeton University Press, 2011.
[6] 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.
[7] A. Rezaei, S.M. Far, M. Soleymani, Near linear-time community detection in networks with hardly detectable community structure, in: 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), IEEE, 2015, pp. 65–72.
[8] M. Rosvall, C.T. Bergstrom, Maps of random walks on complex networks reveal community structure, Proc. Natl. Acad. Sci. 105 (4) (2008) 1118–1123.
[9] G.K. Orman, V. Labatut, H. Cherifi, Qualitative comparison of community detection algorithms, in: International Conference on Digital Information and Communication Technology and Its Applications, Springer, 2011, pp. 265–279.
[10] L. Danon, A. Diaz-Guilera, J. Duch, A. Arenas, Comparing community structure identification, J. Stat. Mech. Theory Exp. 2005 (09) (2005) P09008.
[11] 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.
[12] N. Dugué, A. Perez, Directed Louvain: Maximizing Modularity in Directed Networks, Ph.D. thesis, Université d'Orléans, 2015.
[13] H.-J. Li, Z. Bu, A. Li, Z. Liu, Y. Shi, Fast and accurate mining the community structure: integrating center locating and membership optimization, IEEE Trans. Knowl. Data Eng. 28 (9) (2016) 2349–2362.
[14] H.-J. Li, Z. Bu, Z. Wang, J. Cao, Y. Shi, Enhance the performance of network computation by a tunable weighting strategy, IEEE Trans. Emerg. Top. Comput. Intell. 2 (3) (2018) 214–223.
[15] Z. Bu, J. Cao, H.-J. Li, G. Gao, H. Tao, Gleam: a graph clustering framework based on potential game optimization for large-scale social networks, Knowl. Inf. Syst. 55 (3) (2018) 741–770.

Please cite this article in press as: X. Li, Directed LPA: Propagating labels in directed networks, Phys. Lett. A (2018), https://doi.org/10.1016/j.physleta.2018.11.047