Neurocomputing 151 (2015) 1063–1075

Contents lists available at ScienceDirect

Neurocomputing

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

# Detecting community structure via synchronous label propagation

Shenghong Li*, Hao Lou, Wen Jiang, Junhua Tang

*School of Electronic Information and Electrical Engineering, Shanghai Jiaotong University, Shanghai 200240, China*

ARTICLE INFO

*Article history:*
Received 26 November 2013
Received in revised form
9 April 2014
Accepted 28 April 2014
Available online 18 November 2014

*Keywords:*
Community structure
Community detection
Label propagation
Synchronous
Parallel

ABSTRACT

Community detection has become an important methodology to understand the organization and function of various real-world networks. Label propagation algorithm (LPA) is a near linear time algorithm which is effective in finding a good community structure. However, it updates the labels of nodes asynchronously in random order to avoid label oscillations, resulting in poor performance, weak robustness and difficulty in parallelizing the update procedure for large-scale network and distributed dynamic complex networks. We propose a novel strategy named LPA-S to update the labels of nodes synchronously by the probability of each surrounding label, which is easy to be parallelized. We experimentally investigate the effectiveness of the proposed strategy by comparing with asynchronous LPA on both computer-generated networks and real-world networks. The experimental results show our LPA-S does not harm the quality of the partitioning while can be easily parallelized.



## 1. Introduction

Many complex systems, such as social communication network, biological interaction network and Internet, can be modeled as networks with vertices for individuals and edges for relations between them. Community structure is a significant property of complex networks as it often represents organized groups or functional modules with nodes of common features and accounts for the functionality of the system. Therefore, the capability of detecting community structure provides an important insight into the organization and functional behavior of the real-world system.

A community in a network is usually a group where nodes are densely interconnected and sparsely connected to other parts of the network [1–3]. The problem of community detection, also called community clustering, is quite challenging and has garnered ample interest in the past decade. In the literature, several different approaches have been proposed to find community structure in networks. In Refs. [4–6], a spectral bisection method exploited the spectral property of the Laplacian matrix or normal matrix to divide a network into two groups so that the number of edges between groups is minimized. Kernighan–Lin algorithm in Ref. [7] attempted to minimize the difference between intra-connected edges and inter-connected edges to detect communities. Newman and Girvan first introduced the quality function *Modularity Q* to define a stop criterion for the algorithm in Ref. [8] to detect community. And since then *Modularity* has rapidly become an essential element of many clustering methods. Algorithms like greedy optimization in Ref. [8], simulated annealing in Ref. [9], extremal optimization in Ref. [10] and spectral optimization in Ref. [11], try to optimize the modularity function to detect community structure. Betweenness-based algorithm proposed by Girvan and Newman [2] removes the edges which have maximum betweenness to split the network to generate a dendrogram. CPM algorithm in Ref. [12] can identify overlapping communities based on the assumption that a community is composed of a number of adjacent *k*-cliques. Structural algorithm [13] and dynamic algorithm [14] by Rosvall and Bergstrom turned the problem into optimally compressing the information on the structure of the graph, so that one can recover as closely as possible the original structure when the compressed information is decoded. They are indeed effective in finding a good community structure. Potts model approach [15] by Ronhovde and Nussinov is based on the minimization of the Hamiltonian of a Potts-like spin model, where the spin state represents the membership of the node in a given community. The method is rather fast and its complexity is slightly super-linear.

However, most algorithms mentioned above are limited when used in very large scale networks because of high time complexity or no priori knowledge. For example, the GN algorithm [2] requires that a centrality score is computed for each edge. This centralized and global control for computing centrality scores is a significant bottleneck in speed boost. Some algorithms with linear time complexity have been proposed. Wu and Huberman [16] proposed WH algorithm based on notions of voltage drops to find communities in linear time. Still WH algorithm needs much priori information such as community number, community size and "pole" nodes, making this algorithm difficult to apply when no priori information is available. Label propagation algorithm (LPA) proposed by Raghavan et al. [17] shows good prospect. It employs

* Corresponding author.
*E-mail address:* shli@sjtu.edu.cn (S. Li).

http://dx.doi.org/10.1016/j.neucom.2014.04.084
0925-2312/

the diffusion of information to identify communities. Every node is initialized with a unique label and at every step each node adopts the label that the most of its neighbors currently have. This agent-based and decentralized approach brings near linear time complexity, requires no priori information and is suitable for large-scale networks with millions of nodes and edges [18].

Nevertheless, LPA basically updates the label of nodes asynchronously to ensure convergence. The random asynchronous update order leads to poor performance, weak robustness, and especially difficulty in parallelizing the update procedure. Since parallelism is important for very large scale network and distributed dynamic complex network in which global information is difficult to be collected accurately and timely, in this paper we propose a synchronous version of LPA, namely LPA-S, to achieve this goal. The features of LPA-S include (1) the label of a node is updated synchronously. This synchronous update process can be easily parallelized. (2) A probabilistic update strategy is designed to avoid label oscillation in synchronous label update. Each label is assigned a dynamic probability and the label of a node is updated stochastically according to the probability of each surrounding label. (3) In label probability calculation, we introduce $t$-level *familiarity* function strategy to get better performance.

The remainder of the paper is organized as follows. In Section 2, we introduce the basic label propagation algorithm. Our proposed new algorithm is described in Section 3. The experimental results are presented in Section 4, and Section 5 gives our conclusion.

## 2. Label propagation algorithm

A complex network is represented by $G(V, E)$, where $V$ is the vertex set and $E$ is the edge set. In this paper, we focus our attention to unweighted, undirected networks. Each node $v$ ($v \in V$) has a label $c_v$. $N(v)$ is the set of neighbors of node $v$. The basic label propagation algorithm (LPA) initializes every node with a unique label which will be used to determine the community it belongs to. Nodes update their labels step by step. At every step each node updates its original label to the label shared by the maximum number of its neighbors, i.e.

$$c_v = \arg\max_l N^l(v) \tag{1}$$

where $N^l(v)$ is the set of neighbors of node $v$ that have the label $l$, and $|X|$ is the cardinality of set $X$. By this iterative process densely connected groups of nodes form consensus on one label to form communities. Finally, LPA converges when no node changes its label anymore. Nodes sharing the same label are classified into the same community.

LPA exhibits near linear time complexity $O(km)$, where $k$ is the number of iteration and $m$ is the number of edges. Raghavan et al. [17] mentioned that 95% of nodes or more are classified correctly by the end of iteration 5. The expected number of iterations grows logarithmically with respect to the size of network empirically as reported in Ref. [19]. However, determining the speed of basic LPA is still an open problem. The speed of asynchronous LPA is not sufficient for real time application in very large scale network. Also it is not suitable for distributed dynamic complex networks. Reconsider the basic LPA, we can find that the basic LPA compromises when choosing the update model between synchrony and asynchrony. Because of oscillation of labels, it adopts the asynchronous update. This erases one of the biggest advantages this agent-based algorithm may bring, i.e. parallelism which is important when real-time response is needed in large scale network and distributed dynamic complex network.

Hence synchronous update is necessary. Cordasco and Gargano present a semi-synchronous label propagation algorithm in Ref. [20]. But this algorithm is semi-synchronous and cannot be easily paralleled. Our paper focuses on this point and proposes a novel strategy to achieve synchronous update.

## 3. Synchronous label propagation

### 3.1. Oscillation and probabilistic update

As suggested in Ref. [17], certain properties may prevent the update procedure from converging. Leung et al. claim in Ref. [19]

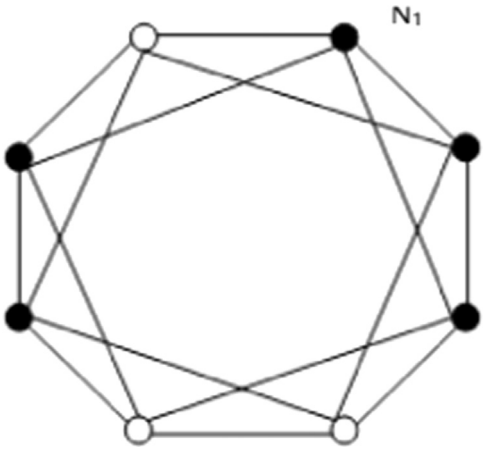


**Fig. 2.** A new configuration that jumps out the label ring since node $N_1$ stay original black label.

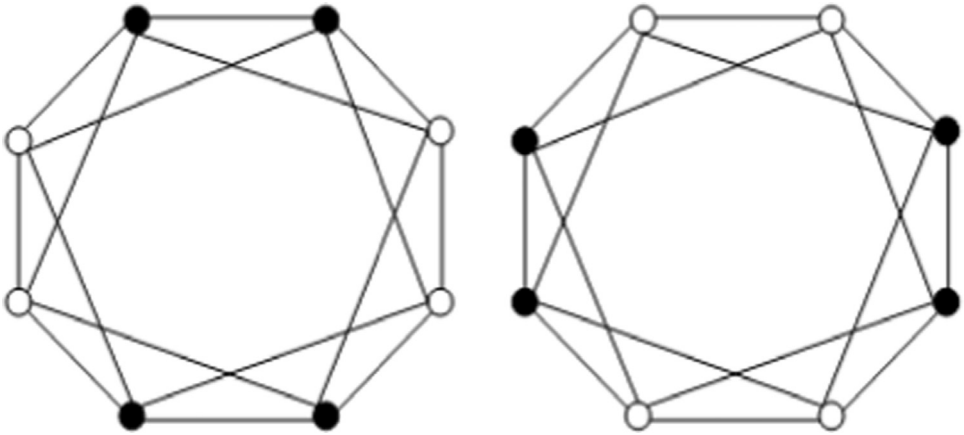

**Fig. 1.** The oscillation phenomenon on a non-bipartite network. Once one of the two configurations is entered, the labels indefinitely oscillate between them.