Community Detection Using Robust Label Propagation Algorithm

Lixin Shi, Junxing Zhang
College of Information and Communication Engineering, Dalian Nationalities University,
116600, China
Tel.: +86-411-87656330
E-mail: shilixin@dlnu.edu.cn

Received: 17 October 2013 / Accepted: 9 January 2014 / Published: 31 January 2014

Abstract: Because there is so much randomness, the robustness of label propagation algorithm (LPA) is severely hampered. To reduce the randomness, a label propagation algorithm based on degree (LPAD) is proposed. Only the node with extreme degree is labeled initially, and the label is updated according to the sum degree of its neighbors during iteration. The results show that the randomness of LPAD is reduced greatly, the robustness of community partition is improved significantly, and the convergence is speeded up too.

Keywords: Community detection, Label propagation algorithm, Degree, Randomness, Robustness.

1. Introduction

Complex networks touch upon many fields from natural science to social science. Common features of complex networks are small-world effect, power-law degree distributions and community structure. Community structure plays important roles in different real-world systems. For instance, it can reveal the social groups with common interests, hobbies or social contexts in social networks [1], can discover the structure units with relevant function in protein networks [2], can improve searching efficiency and correction in World Wide Web [3].

So many algorithms that find reasonably good quality communities have been proposed in recent years. GN algorithm can divide network into any number of communities based on the iterative removal of edges with high “betweenness” scores, whose time complexity is $O(nm^2)$ [4]. NM is an agglomerative algorithm repeatedly merging smaller groups according to modularity, which runs in time $O((m+n)n)$ [5]. Raghavan et al. propose a method called label propagation algorithm (LPA) to identify community in large networks, which runs at nearly linear time complexity [6].

As a result of the simplicity and high efficiency, LPA becomes the first choice algorithm for large real-world networks and is further improved by various authors. Barber and Clark reformulate the LPA as an equivalent optimization problem, giving an objective function whose maxima correspond to community solutions [7]. Combining modularity-specialized LPA and multi-step greedy agglomerative algorithm, Liu and Murata propose an advanced modularity specialized label propagation algorithm [8]. Leung et al. extend LPA by incorporating heuristics like hop attenuation score to improve the quality of the detected communities [9]. Despite various subsequent advances, an important issue of the LPA has not been properly addressed. Because of introduction of randomness,
the robustness of the algorithm is severely hampered, and consequently the stability of the identified community structure is hampered too. To acquire stable community structure, Kato, Kashima and Sugiyama present a new robust transductive learning method that makes predictions by integrating different networks [10]. Šubelj and Bajec propose a balanced propagation that counteracts for the introduced randomness by utilizing node balancers [11]. There are some improved algorithms, whose stability is enhanced, but complexity is increased. These algorithms lost the core advantage of LPA, which is nearly linear time complexity. In this paper, we analyze the randomness of LPA, and propose a degree based label propagation algorithm. The randomness of algorithm is reduced, and the robustness of community structure is improved without additional time cost.

The remainder of this paper is organized as follows: Section 2 reviews the basic LPA, and analyzes its drawbacks. Section 3 introduces degree based label propagation algorithm, and discusses its randomness. Section 4 reports the experimental results. Our conclusions are discussed in Section 5.

2. The Randomness of LPA

Social and other networks are conveniently described as a graph \( G(N,E) \), where \( N \) is the set of nodes and \( E \) is the set of edges. Let \( \omega_{nm} \) be the weight of the edge incident to nodes \( n,m \in N \), and let \( c_n \) denote the label of node \( n \in N \). Let \( N(n) \) denote the set of its neighbors.

The label propagation algorithm is conceptually simple in its procedure. First, initialize the labels at all nodes in the network. Each node \( n \in N \) is labeled with unique label, \( c_n = l_n \). Then, at each step of iteration, each node makes its own decision to change its label to the one carried by the largest number of its neighbors.

\[
c_n = \arg \max_{l \in L} \sum_{m \in N(n)} \omega_{nm},
\]

where \( N(n) \) is the set of neighbors of \( n \in N \) that share label \( l \). As the labels propagate, densely connected groups of nodes quickly reach a consensus on a unique label. When many such dense groups are created throughout the network, they continue to expand outwards until it is possible to do so. Finally, when the algorithm converges, nodes having the same labels are grouped together as one community. Due to extremely fast structural inference of label propagation, the algorithm exhibits near linear time complexity.

Nevertheless, there is so much randomness in LPA, which is damaging to the stability of algorithm. Studying the procedure of LPA, it is known that the randomness comes from two aspects below.

2.1. Label Updating Order

The example network consists of two communities in Fig. 1. Initialize \( c_1=l_1, c_2=l_2 \), and so on.

2.2. Label Selecting

When node \( n_1 \) is updated first, it will adopt label \( l_1 \) or \( l_2 \). No matter which label is updated next, at the end of this iteration, \( n_1, n_2 \) and \( n_3 \) will be labeled with the same label, which is corresponding to natural community \( c_1 \). Because of symmetry, when node \( n_2 \) is updated first, all three nodes will be labeled with the same label at the end of iteration. On the other hand, when node \( n_3 \) is updated first, it can adopt the label \( l_1 \) or \( l_2 \), the algorithm proceeds similar as above. Unfortunately, it can also adopt the label of node \( n_4 \), which is part of community \( c_2 \). Then, after node \( n_1 \) and \( n_2 \) are updated, all three nodes will be labeled with the label \( l_1 \), naturally, and will be classified to community \( c_2 \). If the core node of community is updated first, LPA will classify community right. Otherwise, when the border node between two communities is updated first, the algorithm will lead to wrong classification. Obviously, the algorithm is extremely sensitive to updating order.

3. LPA Based on Degree

Since randomness severely affects the stability, it should be avoided or reduced. First, how to reduce
the randomness of updating order should be considered. If the node is simply updated in some predefined order, the stability will be clearly improved. Although none of the deterministic order performs well in all networks, there are always some orders for specific network which lead to right and stable classification. It is pointed out in Section 2 that the node lying in the center of community should be updated early, while the node between communities should be updated late. Second, to reduce the randomness of label selecting, the inherent property should be considered. Therefore, the degree of node is introduced, which describes the connection of node, and can be applied to distinguish center node from border node.

3.1. The Description of Algorithm

The LPA based on degree (LPAD) can be described in the following steps.

Step 1: Calculate the degree \( D(n) \) of every node.
Step 2: Search the local extreme degree. If \( D(n) \geq D(m) \) for \( \forall m \), then \( n \) is an extreme node. Where \( m \in N(n) \), and \( N(n) \) is the set of neighbor of \( n \).
Step 3: Initialize the extreme node only. For a given extreme node \( x \), let \( C_4(O)=x \). Leave the remaining nodes unlabeled.
Step 4: Set iteration initial value \( t=1 \).
Step 5: Set the updated number \( num=0 \).
Step 6: Choose each node randomly, and update its label to the one carried by the largest sum of degree among its neighbors.

\[
c_u = \arg \max_l \sum_{m \in N(x)} D_n
\]

When the label changes, let \( num=num+1 \).
Step 7: If \( num>0 \), set \( t=t+1 \) and go to Step 5. Else, stop the algorithm.

The algorithm will take additional time to calculating the degree and searching the extreme node comparing with LPA. In Step 1, calculating all degree requires \( O(m) \) time. In Step 2, searching the local extreme node requires \( O(m) \) time too. Besides, there is not any additional time cost. That is to say, the time complexity of LPAD algorithm is still \( O(m) \).

3.2. The Randomness of Algorithm

The updating order of basic LPA is thoroughly random, as well as the label selecting. In view of the above situation, we have taken two steps to reduce the randomness.

3.2.1. Initialize the Extreme Node Only

Only the extreme node is initialized label, while other nodes are not initialized. At the preliminary stage of iteration, if the node selected randomly is near the extreme node, there is almost only the label of the extreme node among all the neighbors. Thus, the node can update its label to that label inevitably. Conversely, if the node selected randomly is far away from the extreme node, there is not any label among its neighbors, the node can not update its label. At the later stage of iteration, there are labels at the extreme node itself, as well as at the closest node. In this case, the node nearby all labeled node can update its label. That is to say, although the node selecting is random, the label updating order is roughly orderly, from the community center gradually to community border.

3.2.2. Select Label According to the Sum of Degree

Among the neighbors of selected node, there may be two or more labels. In this case, the selected node updates its label according to Eq. (2). There are densely connections within community, and loosely connections between communities. The degree of node is higher at the center of community than at the border of community. Thus, the selected node prefers to select the label closer to the center of community. Because of calculating the sum of degree, not simply the number of the neighbors with the same label, neighbor group barely has the same sum of degree each other. Therefore, randomness of the algorithm is greatly reduced, and the stability is obviously improved.

4. Experiments

4.1. Performance Metrics

In order to measure the effectiveness of our community detection algorithm, variation of information (VOI) is used in the experiments [11].

\[
VOI = H(C|P) + H(P|C),
\]

where \( H(C|P) \) is the conditional entropy. The scope of \( VOI \) value is from zero to \( \log N \). For meaningful comparisons, the \( VOI \) value is usually normalized. The lower value represents better correlation between partitions.

4.2. Results and Discussion

4.2.1. Dolphins Network

LPAD is used to analyze on dolphins network first. There are 62 nodes in the network which
represent the dolphins and 159 edges which represent associations between dolphin pairs occurring more often than expected by chance. The network was studied by the biologist David Lusseau, who divided the dolphins in two groups according to their age.

The distribution of degree is shown in Fig. 2, where there are five extreme nodes, \( n_7 \), \( n_{12} \), \( n_{35} \), \( n_{42} \) and \( n_{48} \). Only these extreme nodes are initialized label, \( l_7 \), \( l_{12} \), \( l_{35} \), \( l_{42} \) and \( l_{48} \), while other nodes are not initialized.

**4.2.1.1. Randomness**

To avoid the influence of accident, LPAD is applied to dolphins network 1000 times. Although there is label updating during every iteration, the following analysis shows that label updating mostly occurs during the first iteration, and it is predominant. In this proposition the updating order of each node is show in Fig. 3. The label updating order is roughly orderly even though the node selecting is thoroughly random. For instance, during the first iteration, nodes \( n_4 \), \( n_5 \) and \( n_6 \) which are adjacent to the extreme node \( n_7 \) can update their labels at any time, while nodes \( n_2 \) and \( n_3 \) which are one hop to the extreme node can update their labels only after one of nodes \( n_4 \), \( n_6 \), \( n_8 \) and \( n_9 \) has updated their labels, furthermore, node \( n_7 \) which is two hops to the extreme node can update its label only after node \( n_2 \) has updated its label. It is shown that the updating order of nodes \( n_4 \), \( n_5 \) and \( n_6 \) is from 1 to 54, and concentrates on from 1 to 10, while that of nodes \( n_2 \) and \( n_3 \) is from 3 to 55, concentrates on from 25 to 48, and that of node \( n_7 \) is from 13 to 53, concentrates on from 44 to 50. It can be illustrated more clearly from the average updating order (blue line in Fig. 3).

**4.2.1.2. Stability**

Within 1000 times tests, a series of stability indexes is calculated, which is shown in table 1. In order to compare the performance of LPAD, the same series of LPA and BPA is shown in Table 1 too. The sort of community partition is counted into distinct community partitions. The bigger the number is, the more dispersive the pattern of community partition is, the lower the stability of algorithm is. Comparing distinct community partitions of three algorithms that of LPA is 153, that of BPA is 36, and that of LPAD is 11. The concentration degree of LPA is lowest, and that of LPAD is highest. Comparing community number and community scale, there is very small community which contains only two nodes when LPA is applied to the network, and these is very big community containing all nodes when BPA is applied. Since only the extreme node is initialized label when LPAD is applied, and new label can not been generated during iteration, the final community number must less than or equal to the number of extreme node. In addition, when the node at the border of community selects its label, big community has more attractive than small community. Thus, very small community gradually merges with adjacent community, while two equivalent communities keep balance to prevent
merging into one very big community. Applying LPAD to the network, the community number is from 2 to 4, and the community scale is from 5 to 46.

As a benchmark dataset, the community partition of dolphins network is well known. On the basis of accepted community partition, the variation of information (VOI) is calculated. The VOI of LPA is 0.252, which is a little big. The VOI of BPA and LPAD are 0.085 and 0.067 respectively. The randomness of label selecting of BPA is smaller than that of LPAD, but the node updating order is thoroughly random. By contrast, there is still some randomness of label selecting of LPAD, but the label updating order is roughly orderly, from the community center gradually to community border. In general, the stability of LPA is worst, that of BPA is better, and the stability of LPAD is best.

Table 1. Comparison of three algorithms on stability.

<table>
<thead>
<tr>
<th></th>
<th>LPA</th>
<th>BPA</th>
<th>LPAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distinct community partitions</td>
<td>153</td>
<td>36</td>
<td>11</td>
</tr>
<tr>
<td>Range of community number</td>
<td>2-6</td>
<td>1-5</td>
<td>2-4</td>
</tr>
<tr>
<td>Average community number</td>
<td>3.63</td>
<td>2.45</td>
<td>2.53</td>
</tr>
<tr>
<td>Range of community scale</td>
<td>2-48</td>
<td>3-62</td>
<td>5-46</td>
</tr>
<tr>
<td>Average community scale</td>
<td>17.21</td>
<td>25.31</td>
<td>21.92</td>
</tr>
<tr>
<td>Variation of information (VOI)</td>
<td>0.252</td>
<td>0.085</td>
<td>0.067</td>
</tr>
</tbody>
</table>

4.2.1.3. Convergence

As each iteration of LPA needs $O(n)$ time, the speed of algorithm depends on the number of iterations. The number of iterations of LPA, BPA and LPAD are shown in Fig. 4, which are from 5 to 50, from 4 to 12 and from 4 to 8 respectively. The number of LPA which often exceeds 50 is very dispersive, while those of BPA and LPAD which fluctuate within a narrow range are relatively centralized. Within 1000 times tests, the average number of LPA is the most, that of BPA is less, and that of LPAD is the least, which are 10.8, 6.37 and 4.44 respectively.

In addition, the label updating ratio of each iteration is calculated within 1000 times tests, which is shown in Fig. 5. Regardless of LPA, BPA or LPAD, most labels are updated during the first iteration, which is predominant. The number of updated labels is gradually reduced in subsequent iterations, which is inversely proportion to iteration times. Therefore, only the updating order during the first iteration is calculated. During first 10 iterations, the sum of updating labels of BPA is the most, which is 159.03 % to the number of nodes that of LPA is close to that of BPA, which is 156.92 %, and that of LPAD is the least, which is only 110.61 %. Almost one third updating labels of BPA or LPA are repeated, while only one tenth of LPAD needs to repeat.

Fig. 4. Number of iterations (blue line corresponds to LPA, green line corresponds to BPA, red line corresponds to LPAD).

Fig. 5. Updating label ratio.

4.2.2. Other Benchmark Network

LPAD is used to analyze on other benchmark network too, including: i) karate club: there are 34 nodes which represent club member and 78 edges which represent friendship between members; ii) american football: there are 115 nodes which represent football team and 616 edges which represent competition between teams; iii) jazz musicians: there are 198 nodes which represent musician and 2742 edges which represent partnership between musicians; iv) world wide web: there are 578 nodes which represent web page and 1518 edges which represent link between pages.

In order to compare the performance, LPA and BPA are applied on the above networks 1000 times
Distinct community partitions, variation of information and average iteration times are shown in table 2, 3 and 4 respectively. The experiments reveal the same results as dolphins network. The distinct community partition of LPA is most dispersive, that of BPA is more concentrative than that of LPA, and that of LPAD is most concentrative. The VOI of LPA is biggest, that of BPA is much smaller than that of LPA, and that of LPAD is smallest. That is to say, the stability of LPA is worst, that of BPA is improved significantly, and that of LPAD is enhanced further. Moreover, the iteration times of BPA is obviously less than that of LPA, and that of LPAD is less than that of BPA too.

### Table 2. Distinct community.

<table>
<thead>
<tr>
<th>Network</th>
<th>LPA</th>
<th>BPA</th>
<th>LPAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>karate</td>
<td>177</td>
<td>23</td>
<td>6</td>
</tr>
<tr>
<td>football</td>
<td>411</td>
<td>164</td>
<td>28</td>
</tr>
<tr>
<td>jazz</td>
<td>65</td>
<td>24</td>
<td>13</td>
</tr>
<tr>
<td>web</td>
<td>328</td>
<td>117</td>
<td>32</td>
</tr>
</tbody>
</table>

### Table 3. Variation of information.

<table>
<thead>
<tr>
<th>Network</th>
<th>LPA</th>
<th>BPA</th>
<th>LPAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>karate</td>
<td>0.244</td>
<td>0.199</td>
<td>0.108</td>
</tr>
<tr>
<td>football</td>
<td>0.095</td>
<td>0.092</td>
<td>0.067</td>
</tr>
<tr>
<td>jazz</td>
<td>0.107</td>
<td>0.045</td>
<td>0.039</td>
</tr>
<tr>
<td>web</td>
<td>0.123</td>
<td>0.083</td>
<td>0.064</td>
</tr>
</tbody>
</table>

### Table 4. Average iteration times.

<table>
<thead>
<tr>
<th>Network</th>
<th>LPA</th>
<th>BPA</th>
<th>LPAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>karate</td>
<td>6.8</td>
<td>5.5</td>
<td>3.0</td>
</tr>
<tr>
<td>football</td>
<td>9.23</td>
<td>7.7</td>
<td>3.8</td>
</tr>
<tr>
<td>jazz</td>
<td>12.62</td>
<td>9.0</td>
<td>5.9</td>
</tr>
<tr>
<td>web</td>
<td>11.15</td>
<td>6.9</td>
<td>5.1</td>
</tr>
</tbody>
</table>

5. Conclusions

On the basis of deeply analyzing the randomness of LPA, a degree based label propagation algorithm is proposed. Compared to basic LPA, there are two aspects being adjusted: i) only the extreme node is initialized label; ii) select label according to the sum of degree during iteration. Although the node selecting is random still, the label updating order is roughly orderly, from the community center gradually to community border. Since neighbor group barely has the same sum of degree each other, the label barely needs to select randomly. The benchmark networks experiments show that the randomness of LPAD is reduced significantly, the stability is improved obviously, and the convergence is sped up somewhat. In general, LPAD retains the simplicity of LPA, but improves on its stability.

**References**