Community Detection in Real Large Directed Weighted Networks

Yao Liu, Gang Liu, Qiao Liu, Zhiguang Qin

School of Computer Science and Engineering,
University of Electronic Science and Technology of China
E-mail: liuyao@uestc.edu.cn

Abstract

In this paper, the impact factors of edge weight and vertex weighted degree are introduced into community detection, and the directed weighted degree is used to measure the importance of the node. Based on the modularity optimization, a new community detecting algorithm with flexible and multi-task architecture for directed and weighted networks is proposed. Then the community detection on the real large mail network is conducted, and the experimental results demonstrate that in large directed and weighted networks, the proposed algorithm is efficient with shorter execution time. By comparing the detecting results under different network types, we conclude that the use of weights can improve partition effectiveness and accuracy.

Keywords: Community Structure, Edge Weight, Vertex Weighted Degree, Modularity

1. Introduction

Many complex systems in nature and society can be described in terms of networks or graphs—collections of vertices joined in pairs, for example, the Internet, the world-wide-web, the social and biological systems of various kinds, and so on. Each of these networks consists of a set of nodes or vertices. The nodes or vertices represent computers or routers on the Internet or people in social networks, connected together by links or edges. The links and edges represent data connections between computers or friendships between people. Although a lot of studies have been carried out by mathematics and sociology, physicists show great interest in this field recently and bring new ideas and methods to analyze and model the system [1].

Complex networks are usually characterized by several distinctive properties: power law degree distribution, short path length, clustering and community structure. One network feature that has been emphasized in recent work is community structure, the division of network nodes into groups within which the network connections are dense, but between which they are sparser. The ability to find and analyze such groups can provide invaluable help in understanding and visualizing the structure of networks.

2. Prior work

The study of detecting community structure in networks has a long history. It is closely related to the ideas of graph partitioning in graph theory and computer science, and hierarchical clustering in sociology [2]. Before presenting our own findings, it is worth reviewing some of the preceding work, to understand its achievements and where it falls short.

Graph partitioning is a problem that arises in parallel computing. It has been based on iterative bisection. We find the best division is to divide the complete graph into two groups, and then further subdivide those two until we get the required number of groups. Among the algorithms for the problem, two have dominated the literature: the spectral bisection method [3, 4], which is based on the eigenvectors of the graph Laplacian; and the Kernighan–Lin algorithm [5], which uses a greedy algorithm to improve the network division by optimizing the edges’ number within-communities and between-communities.

Sociologists, in their study of social networks, have developed a substantial body of wisdom about the interpretation and analysis of graphs. The principal technique in current use is hierarchical clustering [6]. These techniques are aimed at discovering natural divisions of (social) networks into groups, based on various metrics of similarity or strength of connection between vertices. They fall into two broad classes, agglomerative and divisive [7], depending on whether they focus on adding edges to
the network or removing edges from the network.

To combat this problem, a number of new algorithms have been proposed in recent years. Girvan and Newman [2, 8] proposed a divisive GN algorithm that uses edge betweenness as a metric to identify the communities boundaries. This algorithm has been applied successfully to a variety of networks, but makes heavy demands on computational resources, running in $O(mn)$ time on an arbitrary network with $m$ edges and $n$ vertices, or $O(n^3)$ time on a sparse graph. This restricts the algorithm’s use to the networks with thousands vertices under current hardware condition.

More recently a number of faster algorithms have been proposed [9-13]. In [9], an agglomerative algorithm based on the greedy optimization of the modularity [2] quantity was proposed. This method appears to work well both in contrived test cases and real-world situations. It is substantially faster than the Girvan and Newman algorithm. The faster Newman algorithm runs in time $O((m+n)n)$, or $O(n^3)$ on a sparse graph. In [14], Clauset and Newman presented the CNM algorithm that performs the same greedy optimization as the faster Newman algorithm. By using the heap data structure, the CNM algorithm runs far more quickly, in time $O(nd\log n)$ where $d$ is the depth of the dendrogram. If the network is sparse $m/n$, then the running time is essentially linear, $O(n\log n)$. In [15], MSG-MV algorithm, a multistep extension of the greedy algorithm, optimized modularity by an iterative procedure in which multiple pairs of communities are merged at each iteration.

3. Our algorithm in weighted networks

The modularity optimization algorithms are the most widely used algorithms. These algorithms described in section 2 are all belong to unweighted networks. However, many networks are intrinsically weighted, their edges have different strengths. A single line representing the existence of the relation will be a limitation when it is used to describe more than one level relation. So to fully characterize the interactions in real networks, weight of links should be taken into account.

Edge weights in networks have received relatively less attention in the physics literature for the reason that in any field one is advised to look at the simple cases first (unweighted networks) before moving on to more complex ones (weighted networks) [1]. In this paper, we investigated a campus mail network which comprised more than 5000 mail users, and proposed our modularity optimization algorithm with multi-task architecture in the directed weighted network to detect community structure.

3.1. Measurements of weighted network

A weighted network can be represented mathematically by an adjacency matrix with entries that are not simply zero or one, but are equal instead to the weights on the edges:

$$W_{ij} = \text{weight of connection from } i \text{ to } j$$

The edge weight is used to measure the correlation of the edge from vertex $i$ to vertex $j$. The higher of the $W_{ij}$ value, the greater possibility for the information transmission from $i$ to $j$ which means the two nodes are connected closely. On the contrary, the information transmission from $i$ to $j$ is difficult, which means the two nodes are connected sparsely.

The vertex weighted degree $w_i$ of vertex $i$ in a weighted network is the summation of edge weight attached to it:

$$w_i = \sum_j W_{ij}$$

The vertex weighted degree is used to characterize the effectiveness of the node in the network. The value of $w_i$ indicates the importance of the vertex in the network, which means the information is easy to spread. Conversely, the effect of the vertex in the network is relatively small and isolated.

The weighted mail network is defined by considering the strengths of the links in terms of mail numbers between users. The vertices represent the mail users. An edge is drawn between a pair of users...
if they exchanged mails, and the number of mails is the weight of the edge. So, it can be represented by
a weight matrix $W$, where each element $W_{ij}$ stands for the total number of mails from user $i$ to user $j$.

### 3.2. Weighted modularity

In previous work on unweighted networks, Newman solved community problem by introducing a quantity which called the modularity [2, 9, 14]. This quantity is defined as the fraction of edges that fall within communities minus the expected value of the same quantity if edges are assigned at random, conditional on the given community memberships and the degrees of vertices.

$$Q = \text{(fraction of edges within communities)} - \text{(expected fraction of such edges)}$$  \hspace{1cm} (3)

Suppose we have a possible division of a weighted network into communities, let $c_i$ be the community to which vertex $i$ is assigned. Then the fraction of the edges in the graph that fall within communities, i.e., that connect vertices that both lie in the same community, is

$$\frac{\sum_{u,v} W_{uv} \delta(c_u, c_v)}{\sum_{ij} W_{ij}} = \frac{1}{2w} \sum_{ij} W_{ij} \delta(c_i, c_j)$$  \hspace{1cm} (4)

where the $\delta$-function is $\delta(u,v) = 1$ if $u = v$ and 0 otherwise, $W_{ij}$ is the edge weight between node $i$ and $j$, $w_i$ (vertex weighted degree) is the summation of edge weight attaching to node $i$, and $w = \frac{1}{2} \sum_i W_i$ is the summation of edge weight in the network. Thus the modularity $Q^*$, as defined above, is given by

$$Q^* = \frac{1}{2w} \sum_{ij} \left[ W_{ij} - \frac{w_i w_j}{2w} \right] \delta(c_i, c_j)$$  \hspace{1cm} (5)

To simplify the description of the algorithm, let us define the following two quantities:

$$e_{uv} = \frac{1}{2w} \sum_{ij} W_{ij} \delta(c_i, u) \delta(c_j, v)$$  \hspace{1cm} (6)

which is the fraction of edges weights that join vertices in community $u$ to vertices in community $v$, and

$$a_v = \frac{1}{2w} \sum_i w_i \delta(c_i, v)$$  \hspace{1cm} (7)

which is the fraction of ends of edges weights that are attached to vertices in community $v$. Then, writing $\delta(c_i, c_j) = \sum_v \delta(c_i, v) \delta(c_j, v)$, and Eq. (5) can be write as

$$Q^* = \frac{1}{2w} \sum_{ij} \left[ W_{ij} - \frac{w_i w_j}{2w} \right] \sum_v \delta(c_i, v) \delta(c_j, v)$$

$$= \sum_v \left[ \frac{1}{2w} \sum_{ij} W_{ij} \delta(c_i, v) \delta(c_j, v) - \frac{1}{2w} \sum_i w_i \delta(c_i, v) \frac{1}{2w} \sum_j w_j \delta(c_j, v) \right]$$

$$= \sum_v \left( e_{vv} - a_v^2 \right)$$  \hspace{1cm} (8)
where \( e_{uv} = \frac{1}{2w} \sum_i W_i \delta(c_i, u) \delta(c_i, v) \) is the fraction of summation of link weight that connect two nodes inside the community \( v \). Obviously, \( Q^* \) takes both link and link weight into account. It suggests a description for community in weighted networks. We think that weighted community structure is the groups of network vertices. The summation of internal link weight among nodes within groups is larger than that of link weight between groups. In other words, the relations of nodes within group are close, and the relations of nodes between groups are distant. The max value of \( Q^* \) is 1, which indicate strong community structure. In practice, values for such networks typically fall in the range from about 0.3 to 0.7. The peak of the modularity \( Q^* \) corresponds to a perfect identification of the communities.

### 3.3. Our algorithm

Our algorithm performs the greedy optimization algorithm by using the heap data structure in weighted network. In order to deal with large network (i.e. over 20,000 nodes), it is built on a multi-task model, and takes advantage of multi-core processors. The visualization module uses a special 3D render engine to render graphs in real-time. This technique uses the computer graphic card, as video games do, and leaves the CPU free for other computing.

The algorithm uses three data structures:

(a) A sparse matrix containing \( \Delta Q_{uv} \) for each pair \( u, v \) of communities with at least one edge between them. We store each row of the matrix both as a balanced binary tree (so that elements can be found or inserted in \( O(\log n) \) time) and as a maxheap (so that the largest element can be found in constant time).

(b) A max-heap \( H \) containing the largest element of each row of the matrix \( \Delta Q_{uv} \) along with the labels \( u, v \) of the corresponding pair of communities.

(c) An ordinary vector array with elements \( a_i \).

The algorithm can be defined as follows.

1. **Initialization.** We start off with each vertex being the sole member of a community, apparently initial modularity \( Q^* = 0 \). \( e_{uv} = W_{ij}/2w \) if \( u \) and \( v \) are connected and zero otherwise, and \( a_i = w_i/2w \), where \( \delta(c_i, u) = 1 \) and \( \delta(c_i, v) = 1 \). Thus we initially set

\[
\Delta Q_{uv} = \begin{cases} 
W_{ij}/2w - w_i w_j/(2w)^2 & \text{if } u, v \text{ are connected,} \\
0 & \text{otherwise}
\end{cases} \quad (9)
\]

2. Calculate the initial values of \( \Delta Q_{uv} \) and \( a_i \) according to Eq. (9), and populate the max-heap with the largest element of each row of the matrix \( \Delta Q^* \).

3. Select the largest \( \Delta Q_{uv} \) from \( H \), join the corresponding communities \( u \) and \( v \), label the combined community \( v \), update the matrix \( \Delta Q^* \), the heap \( H \) and \( a_i \).

3.1 We need adjust a few of the elements of \( \Delta Q^* \), update the \( v \)th row and column, remove the \( u \)th row and column. (Since the joining of a pair of communities between which there are no edges at all can never result in an increase in \( Q^* \), we need only consider those pairs between which there are edges. The change in \( Q^* \) upon joining two communities is given by \( \Delta Q^* = e_{uv} + e_{wu} - 2a_u a_v = 2(e_{wu} - a_u a_v) \). [9]) The update rules are as follows.

(a) If community \( k \) is connected to both \( u \) and \( v \), then

\[
\Delta Q_{ik} = \Delta Q_{ik}^* + \Delta Q_{vk}^*
\]

(b) If community \( k \) is connected to \( u \) but not to \( v \), then
\[ \Delta Q_{vk} = \Delta Q_{uk} - 2a, a_k \]  

(c) If community k is connected to v but not to u, then

\[ \Delta Q_{vk} = \Delta Q_{uk} - 2a, a_k \]  

3.2 Update the heap H according to \( \Delta Q^v \).

3.3 Update \( a_i : a_i = a_i + a; a_i = 0 \).

3.4 Increase \( Q^v = Q^r + \Delta Q^v \) and record it.

5. Repeat step 3 until only one community remains.

Note that these equations imply that \( Q^v \) has a single peak over the course of the algorithm, since after the largest \( \Delta Q^v \) becomes negative all the \( Q^v \) can only decrease. So when the peak value of \( Q^v \) appears, the algorithm should been stopped and the perfect division is got.

4. Experiment and result analysis

Mail network is an important social network based on the internet. It indirectly reflects people’s social activities by recording mail logs. Based on the analysis of mail network, we can better understand the social network structure and the individual behavior. Mail network is a kind of community net; it is constitutive of some mail communities. Community network can be defined by some method such as relationship network [16].

This paper has analyzed the mail logs of a campus mail system from Jan 1 to Dec 31, 2011. As we all know, there exists a large number of spam mails in the internet which inevitably disturb the data mining research of human social behaviors. The mail server can only completely record the local domain users’ activities, and the users’ activities from outside server cannot be recorded completely. So for the integrity and reliability of the data, we only extract the local users’ mail log. Considering the user’s privacy, we use digital number to replace the mail user’s address. From the mail server, the logs comprise 5,435 local mail users and 1,400,740 mails in 2011 year, and each user averagely sends 257.73 mails per year.

**Definition 4.1.** Mail network is a kind of weighted graph \( G = (V,E,W) \) which \( V \) is the set of nodes, \( E \) is the set of edges, and \( W \) is the weighted function of edges. Node \( v \in V \), represents a mail address (person name or email address). Edge \( e \in E \), \( e = [v_i,v_j] \), represents member \( v_i \) sent a mail to member \( v_j \). \( W_e \) is the number of mails which member \( v_i \) sent to member \( v_j \) as well as the strength of association.

As a matter of convenience, we use \( E \) to represent edge \([v_i,v_j] \).

As we know, if an individual has a lot of direct contacts with others in complex systems, it is more important and has great “power” [17]. Under the guidance of this idea, our algorithm for detecting communities in directed and weighted networks is proposed. Mail network can be seen as a directed graph, member \( v_i \) sends a mail to member \( v_j \), but \( v_j \) may not send a mail to \( v_i \). It means direction of the edges have different influences on the importance of nodes. Thus we measure the importance of a node with directed weighted degree.

**Definition 4.2.** The vertex out-weighted degree is the amount of edge weights leaving the vertex \( v_i \), \( w_{out} = \sum W_{ij} \), the vertex in-weighted degree is the amount of edge weights entering the vertex, \( w_{in} = \sum W_{ji} \), and the vertex weighted degree is defined as \( w_i = w_{out} + w_{in} \).

We remove the system administrator accounts such as “admin”, “webmaster”, “mail-Daemon” or “emdg-daemon”, which always mass mails to all local domain users. We also remove the self-loop edges such as \( E \) in undirected graph. As Fig.1 shown, the mail data set consists of 4,368 active nodes, 49,886 undirected edges with edge weights ranging from 1 to 1,625, and 77,936 directed edges with edge weights ranging from 1 to 876 in one year. The deeper color and the bigger size of the nodes mean the vertex degree is greater.
Modularity described above is right the metric to evaluate the goodness of a partition of a graph. The larger the modularity $Q$ is, the better the partition summarizes the overall community structure[18]. Note that $Q$ is a real number between 0 and 1 and when it is larger than 0.3 the community structure is remarkable[9]. The optimal modularity of real networks usually lies between 0.3 and 0.7. We validate the effectiveness and efficiency of our multi-task algorithms through experiments. The experiments are conducted on Windows xp installed on PC with Pentium Dual-Core 2.5GHZ CPU and 2G MEM.

4.1. Community structure in weighted mail network

To see the effectiveness of our algorithm more deeply, we run different type algorithms on the mail network. Fig. 2 shows the community assignments of the weighted mail network, different colors stand for different communities.

As shown in Table 1, the modularity is not the same in different network types. The lowest value appears in undirected-unweighted type, and the peak value is in undirected-weighted type which indicates the most strong community structure appears in this type. After the use of weights, the modularity increases from 0.6 to 0.7. So we can conclude that the introduction of weights can increase the modularity value, which means it can improve the community detecting performance of the algorithm. We also notice another interesting thing that in weighted condition, the modularity in directed type is lower than in undirected type. The larger number of edges disturbs the community partition performance, which means the use of edge direction cannot improve the detecting performance.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Weighted</th>
<th>Type</th>
<th>Vertices</th>
<th>Edges</th>
<th>Modularity</th>
<th>Communities</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNM</td>
<td>Unweighted</td>
<td>Undirected</td>
<td>4368</td>
<td>49886</td>
<td>0.543</td>
<td>67</td>
</tr>
<tr>
<td>Our Algorithm</td>
<td>Unweighted</td>
<td>Directed</td>
<td>4368</td>
<td>77936</td>
<td>0.609</td>
<td>70</td>
</tr>
<tr>
<td>Our Algorithm</td>
<td>Weighted</td>
<td>Undirected</td>
<td>4368</td>
<td>49886</td>
<td>0.797</td>
<td>72</td>
</tr>
<tr>
<td>Our Algorithm</td>
<td>Weighted</td>
<td>Directed</td>
<td>4368</td>
<td>77936</td>
<td>0.769</td>
<td>71</td>
</tr>
</tbody>
</table>

Figure 2. Community Partition Graph for the Mail Network
We compare the partition results of the four network types in detail. Fig. 3 shows the community partition details, the Y axis represents the vertices number inside each community, the X axis represents the community size rank.

In weighted condition, the community partition results are similar both in directed graph and undirected graph. In unweighted network types, we find super-communities that some communities contain a large fraction of vertices. The biggest community has 624 (14.28%) vertices in undirected-unweighted type, and 712 (16.30%) vertices in directed-unweighted type. On the contrary, the biggest community accounts for only 10% vertices of the total in the weighted types. So the use of weights can prevent the super-communities problem.

So we can summarize that the use of weights can help to increase the partition performance and accuracy of our algorithm. And the use of directions could not improve the partition performance.

![Figure 3. Community Assignments for the Mail Network](image)

**4.2. Statistical results analysis**

In Table 2 we have collected a few interesting statistical properties of the three type networks. The three types’ properties almost have similar values except for the average clustering coefficient and the average (weighted) degree. Clustering coefficient $C_i$ of a node is defined as the ratio of number of links shared by its neighboring nodes to the maximum number of possible links among them. In other words, $C_i$ is the probability that two nodes are linked to each other given that they are both connected to $i$ [19, 20]. The average clustering coefficient is defined as,

$$ C = \frac{1}{N} \sum_{i=1}^{N} C_i $$

(13)

Average clustering coefficient measures the global density of interconnected nodes in the network.

<table>
<thead>
<tr>
<th>Type</th>
<th>Avg.Degree</th>
<th>Avg.Weighted Degree</th>
<th>Avg.Clustering Coefficient</th>
<th>Avg.Path Length</th>
<th>Network Diameter</th>
<th>Graph Density</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directed-Unweighted</td>
<td>17.842</td>
<td>/</td>
<td>0.244</td>
<td>3.36</td>
<td>8</td>
<td>0.04</td>
</tr>
<tr>
<td>Undirected-Weighted</td>
<td>22.842</td>
<td>297.897</td>
<td>0.407</td>
<td>3.282</td>
<td>8</td>
<td>0.05</td>
</tr>
<tr>
<td>Directed-Weighted</td>
<td>17.842</td>
<td>155.515</td>
<td>0.244</td>
<td>3.36</td>
<td>8</td>
<td>0.04</td>
</tr>
</tbody>
</table>

Small-world [21, 22] networks are characterized by a very small average shortest path length and a high average clustering coefficient. The top value of average clustering coefficient is found in undirected-weighted type $C = 0.407$, which is an order of magnitude higher than that of the comparable random network $C_{rand} \sim \langle k \rangle / \langle k \rangle = 0.005$, $\langle k \rangle$ is the average degree. We found the shortest average path
length to be \( L = 3.282 \) in undirected-weighted type, which is an order of same size with that of the random network \( k_{rand} = \ln N / \ln (K) = 2.6792 \). These two properties indicate that mail network is a small-world network. And we can conclude that our algorithm gets the best performance in undirected-weighted type.

5. Conclusion

In this paper we have addressed the topic of weighted network—networks in which the edges between vertices carry weights representing their strength or capacity. Although such networks appear at first to be substantially more difficult to understand than their unweighted counterparts, we have argued that in many cases a mapping of the weighted network onto an unweighted multigraph will allow us to apply directly the results and techniques developed for the unweighted case.

We propose a solution to detect the weighted and directed communities of large mail network, based on modularity method. We analyze the weighted and directed communities, determine the community structure, form and division accurately. The flexible and multi-task architecture which has a considerable speed advantage over previous algorithms brings new possibilities to work with large complex data sets, and the visualization module produces valuable visual results. In the last, we analyze the experiment results in different network types, and found that the use of weights can greatly improve the efficient and accuracy of our algorithm.

6. Acknowledgement

This work was sponsored by the National Natural Science Foundation of China (Grant No. 61133016) and the National High Technology Research and Development Program (“863” Program) of China (Grant No. 2011AA010706).

7. References


