A flexible approach for the hierarchical community detection

Guang-Cao LIU[1,a], Guang ZHAO[1,2], Jiao CHEN[1] and Yu-Lin ZHUANG[1,2]

[1]*Xiamen Great Power Geo Information Technology Co. Ltd., Xiamen, China*
[2]*State Grid Information & Telecommunication Group Co. Ltd., Beijing, China*

Abstract. The hierarchical community detection is able to demonstrate the intrinsic structure of the network, but a deficiency is that the final structure is invariable. In this study, a flexible extension of the hierarchical approach is proposed. The modularity increasement is substituted by a ratio that should be larger than a threshold. We are able to adjust the hierarchical structure by modifying the threshold. The experiments display that such a variation of hierarchical structure is sensible if the modularity metric Q does not change too much.

1 Introduction

The study of community structure detection is a hot topic in both computer science and statistical physics. It is closely related to graph partition, such as the METIS algorithm [1, 2]. Graph partition is able to precisely divide the network into a small number of parts, e. g. 8 parts, in order to facilitate parallel computation, but the results are not close related to the nature structure of the network. By contrast, community detection is able to find the nature community structure of the network, and probably dozens of or hundreds of communities are found. Since there are so many communities, a hierarchical structure is preferred to demonstrate the network, as shown on figure 1. However, the hierarchical community detection is not as flexible as the graph partition that can divide the network into any number. It would be ideal if we can slightly adjust the hierarchical results, especially the number of communities.

The modularity metric Q to evaluate the community structure has been proposed by Newman and Girvan, as shown in equation (1) [3, 4]. The range of Q is between 0 and 1. The large Q indicates an evident community structure. Normally, if Q is below 0.3, the partition of the community structure may not be reasonable.

$$Q = \frac{1}{2m}\sum_{ij}[A_{ij} - \frac{k_i k_j}{2m}]\delta(c_i, c_j) \qquad (1)$$

A_{ij} is the weight of the edge between vertex i and vertex j, $m = \frac{1}{2}\sum_{ij} A_{ij}$ is the sum of weights of all edges, $k_i = \sum_l A_{il}$ is the sum of the weights of the edges attached to vertex i, c_i is the community to which vertex i is assigned, and the $\delta(c_i, c_j)$ is 1 if vertex i and vertex j are assigned to the same community and 0 otherwise.

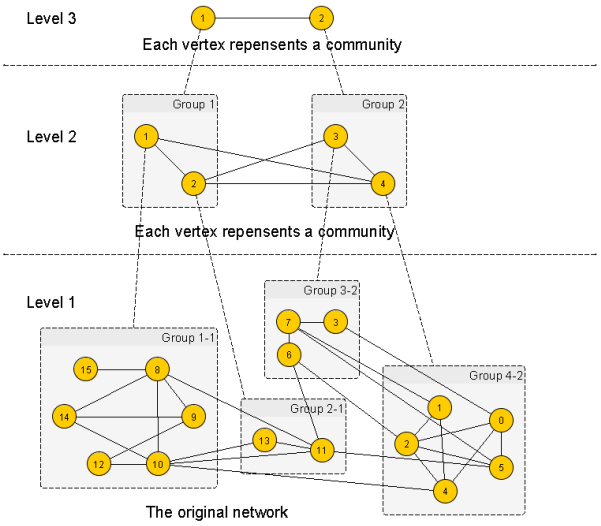

Figure 1. The Example of the Hierarchical Community Structure Detected By the Bgll Method.

There are many community detection methods which either base on or profit from the modularity metric Q [5-8]. Among these methods, a fast and hierarchical approach method has been proposed by Blondel, Guillaume, Lambiotte, and Lefebvre, namely the BGLL method [9]. This method contains two phases that are repeated iteratively. The first phase increases Q by the community adjustment, which tries to move each vertex from its original community and choose one of its neighbour vertex's community to join in so as to maximize Q. The second phase builds a new network which condenses communities found during the first

phase into new vertexes, and new edges are assigned with the sum of weights of previous edges between communities. The two phases are repeated until there is no more changes so that Q cannot be further increased.

2 Method

In the BGLL method, when vertex i is removed from its original community, it re-chooses a connecting community to join in. The increasement of Q, i.e. $\Delta Q_i(C_j)$, for moving vertex i into vertex j's community, is

$$\begin{aligned}\Delta Q_i(C_j) &= \tfrac{1}{2m}[(A_{i,C_j}+A_{C_j,i})-\tfrac{k_i k_{C_j}+k_{C_j}k_i}{2m}] \\ &= \tfrac{1}{2m}[(A_{i,C_j}-\tfrac{k_i k_{C_j}}{2m})\times 2]\end{aligned}, \quad (2)$$

where $A_{i,C_j}=A_{C_j,i}=\sum_{j\in C_j}A_{i,j}$ is the sum of weights of edges that connect vertex i with any vertex in community Cj, and $k_{C_j}=\sum_{j\in C_j}k_j$ is the sum of weights of all edges inside the community Cj.

In all vertexes connected to vertex i, vertex i will choose the maximum one $Max[\Delta Q_i(C_j)]$. If $Max[\Delta Q_i(C_j)]>0$, vertex i joins into the maximum one community Cj, otherwise vertex i stay alone as a community.

In order to adjust the result of the BGLL method, we define a new judegement instead of $Max[\Delta Q_i(C_j)]>0$. The new judgement is

$$\begin{cases} P_i(C_j)=\frac{2mA_{i,C_j}}{k_i k_{C_j}} \\ Max[P_i(C_j)]>P_{thres}\end{cases}, \quad (3)$$

where Pthres is a global threshold that control whether some vertexes can join to their neighbour's communities or stay alone otherwise. The following gives the meaning of this new judgement.

First, the new judgement is an extension of the original BGLL method. When Pthres=1, i.e. the maximum $\frac{2mA_{i,C_j}}{k_i k_{C_j}}>1$, the partition result is the same as the BGLL method, since $\Delta Q_i(C_j)>0$ is equal to $A_{i,C_j}-\frac{k_i k_{C_j}}{2m}>0$ or $\frac{2mA_{i,C_j}}{k_i k_{C_j}}>1$.

Second, $\frac{k_{C_j}}{2m}$ is the ratio of Cj's degrees of links (k_{C_j}) to the sum of degrees of all links (2m), hence $k_i\cdot\frac{k_{C_j}}{2m}$ gives the estimated links from vertex i to community Cj. A_{i,C_j} represents the real links between vertex i and community Cj. Therefore, $\frac{A_{i,C_j}}{k_i k_{C_j}/(2m)}$ is the ratio between the real links and the estimated links. Finally, $\frac{A_{i,C_j}}{k_i k_{C_j}/(2m)}>P_{thres}$ only allows the ratio above the threshold so as to join vertex i into community Cj.

The whole process is carried out as follows.

Step 1. Set the value of Pthres, which is often set to 1 initially.

Step 2. At the beginning of the first phase, assign every vertex a different community. There are as many communities as vertexes.

Step 3. Optimize the belonging community of each vertex.

Step 3.1. Visit all vertexes one by one. A random order is recommended.

Step 3.1.1. Each time pick a vertex out of its original communities, and renew the number of vertexes and edges in its original community.

Step 3.1.2. For the picked out vertex, find out all its adjacent communities, which contain at least one edge connecting to this vertex.

Step 3.1.3. Calculate $P_i(C_j)=\frac{2mA_{i,C_j}}{k_i k_{C_j}}$ for all the adjacent communities found in the above step, in which i denotes the picked out vertex and Cj is an adjacent community.

Step 3.1.4. Choose the maximum one among all of $P_i(C_j)$ obtained above. If the maximum one $Max[P_i(C_j)]>P_{thres}$, vertex i joins into the community Cj, and if community Cj is not i's last community before i is picked out, this is a change in the community structure. On the other hand, if $Max[P_i(C_j)]\le P_{thres}$, vertex i go back to its original community, in which i stay alone without any other vertex in its community. All the values of $Max[P_i(C_j)]$ are recorded as references for tuning Pthres.

Step 3.2. After all vertexes are optimized, if the community structure changes during Step 3.1.4, repeat from step 3.1 to optimize all vertexes again.

Step 4. In the second phase a new network is constructed as follows.

Step 4.1. If any community contains more than one vertex, it is replaced by a new vetex. Otherwise, there is no more than one vertex in every community, and jump to step 6 instead.

Step 4.2. In the new network, new edges are assigned the sum of weights of previous edges between communities. All previous edges inside a commuity generates self loops on the new vertex.

Step 5. Repeat from step 2 to detect communities on the new network.

Step 6. If the detected communities are not desired, adjust the value of Pthres according to recorded $Max[P_i(C_j)]$ in step 3.1.4, and repeat the two phases from step 2.

3 Experiments and results

The proposed method is applied in a Matlab program, which is a modification of the BGLL algorithm originally coded by Antoine Scherrer. Experiments are carried out on many different data sets.

As an example, a small network composed of 20 vertexes are tested. The results are plotted with the yEd graph editor shown on figure 2 to figure 7. For each Pthres, the corresponding modularity Q and the number of top level communities are listed on Table 1. The number of top level communities changes from 7 to 3, but the modularity Q only slightly varies from 0.3682 to 0.3462. Therefore, in such a case, slightly adjust Pthres to change the hierarchical structure is feasible. However, when the number of top level communities is further reduced to 2, the modularity suddenly goes down to Q=0.2949. Hence, it is not suitable to divide the network into only 2 communities due to the low Q value.

Table 1. Results for the Modified P_{thres}.

P_{thres}	Q	Top level communities
1.642	0.3682	7
1.334	0.384	6
1	0.3882	5
0.830	0.3755	4
0.799	0.3462	3
0.590	0.2949	2

Among these variations, some small communities never split in all figures, such as 1 2 3 vertexes, 5 7 9 vertexes, and etc. These invariant small communities validate this method.

The Pthres is modified according to the recorded $Max[P_i(C_j)]$ values in Step 3.1.4. During the community detection process with Pthres=1 on figure 2, all the recorded $Max[P_i(C_j)]$ are {..., 0.7033, 0.800, 0.831, 1.333, 1.641, 2.909, ...}. The closest values to Pthres=1 on both sides are 0.831 and 1.333. Based on the reference to 1.333, Pthres is revised to 1.334 in figure 3. As a result, the number of top level communities grows up to 6. Similarly, based on the reference to 0.831, Pthres is revised from 1 to 0.830 in figure 5, and the number of top level communities decreases to 4.

For each Pthres, the complexity is O[m*log(m)], in which m denotes the number of edges, and the complexity is the same as that of the BGLL method. We have tested the proposed method on networks containing tens of thousands of edges. The BGLL method has been tested on larger networks, such as a network of 1 million edges which could be processed in only 3 seconds [9].

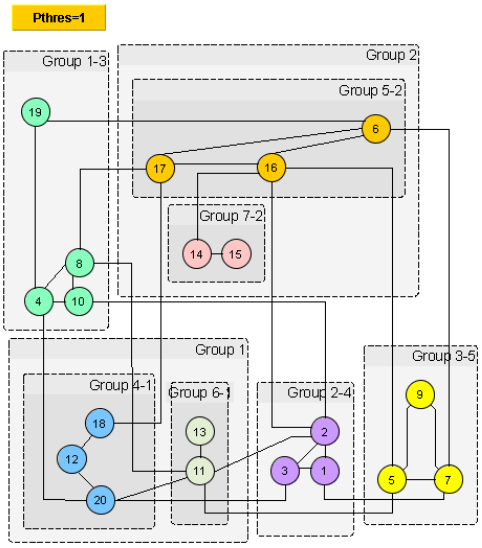

Figure 2. Start from Pthres=1, Q=0.3882, and the Network is Divided to 5 Communities on the Top Level.

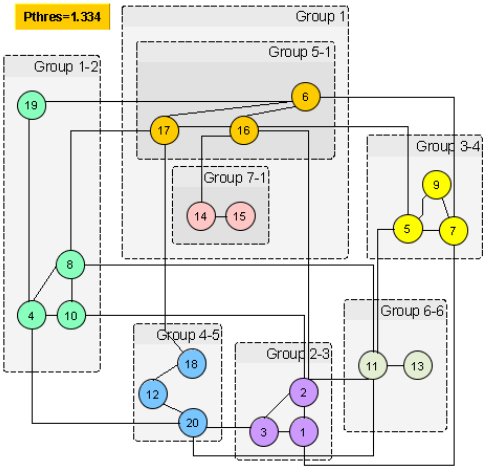

Figure 3. Turn Up the Pthres to 1.334, Q=0.3804. the Number of Communities Grows up to 6 on the Top Level.

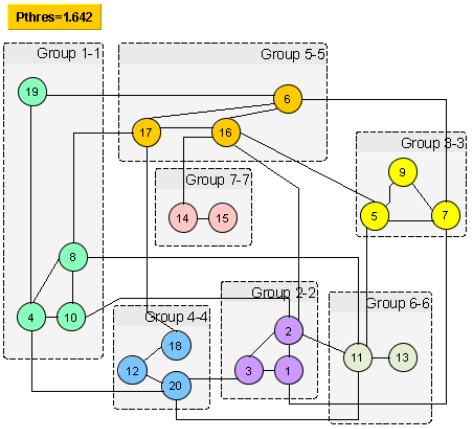

Figure 4. Turn Up the Pthres to 1.642, Q=0.3682. the Number of Communities Grows Up to 7 on the Top Level.

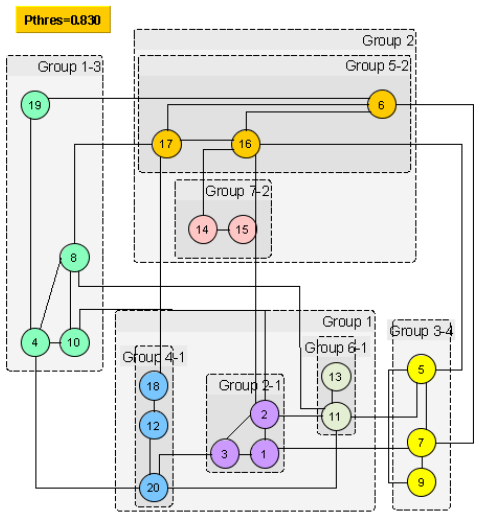

Figure 5. Turn Down the Pthres from 1 to 0.830, Q=0.3755. The Number Of Communities Decreases to 4 on the Top Level.

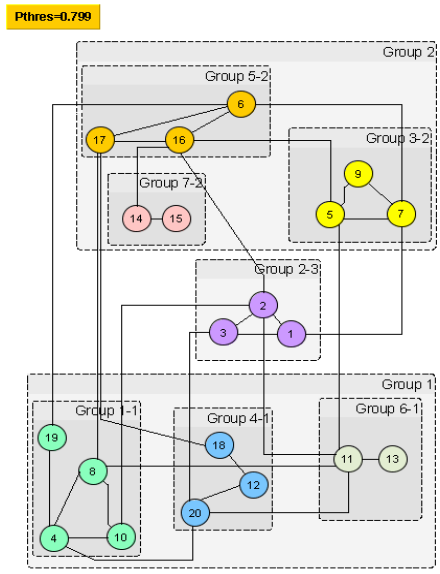

Figure 6. Turn Down the Pthres To 0.799, Q=0.3462. the Number of Communities Decreased to 3 on the Top Level.

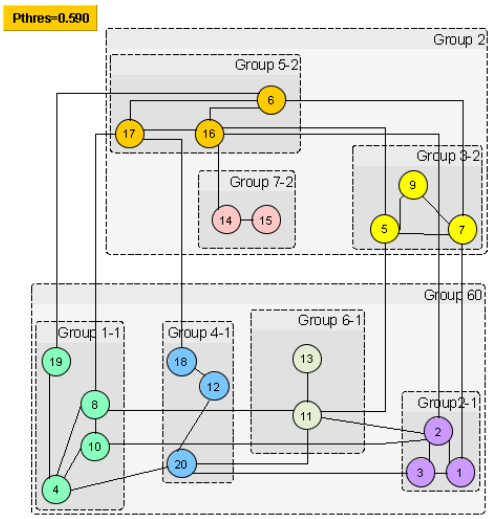

Figure 7. Turn Down the Pthres to 0.590, Q=0.2949. the Number of Communities Decreased to 2 On the Top Level.

4 Discussion

Sometimes setting the Pthres to the closest value according to the recorded $Max[P_i(C_j)]$ may not increase/decrease the number of top level communities, but make a change to the community structure. As an example, there are 3 top level communities on figure 6, and the closest recorded value below Pthres=0.799 is 0.646. If Pthres is set to 0.645, there are also 3 top level communities shown on figure 8, but the structure is different in comparison with figure 6.

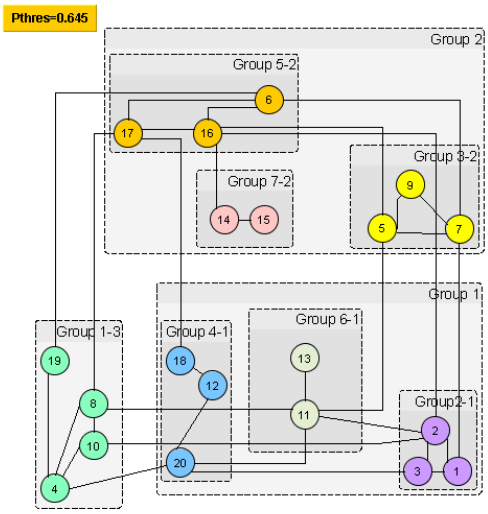

Figure 8. Turn Down the Pthres to 0.799. the Number of Communities Does Not Change on the Top Level in Comparison with Figure 6.

5 Conclusion

We introduce a flexible approach for the hierarchical community detection. The community detection process is an extension of the BGLL method. The flexibility is accomplished according to that $\frac{2mA_{i,C_j}}{k_i k_{C_j}}$ should be larger than a threshold. Experiments prove the availability of this approach.

Acknowledgement

This research is supported by Science and Technology Project of SGCC (Grant No. SGITG-KJ-JSKF[2015]0012).

References

1. G. Karypis, V. Kumar, J Parallel Distr. com. 48, 96 (1998)
2. G. Karypis, V. Kumar, SIAM J. Sci. Comput. 20, 359 (1998)
3. M.E.J. Newman, M. Girvan, Phys. Rev. E 69, 026113 (2004).
4. A. Clauset, M.E.J. Newman, C. Moore, Phys. Rev. E 70 (2004).
5. K. Wakita, T. Tsurumi, *Proceedings of the 16th international conference on world wide web*, 1275 (2007)
6. U.N. Raghavan, R. Albert, S. Kumara, Phys. Rev. E 76, 036106 (2007)
7. R. Ghosh, K. Lerman, Lect. Notes Comput. SC. 5498, 20 (2008)
8. L. Tang, X. Wang, H. Liu, L. Wang, *Workshop on Social Media Analytics*, 14 (2010)
9. V.D. Blondel, J.L. Guillaume, R. Lambiotte, E. Lefebvre, J. Stat. Mech. Theor. Exp. (2008)