

AN INTRODUCTION TO COMMUNITY DETECTION IN GRAPHS

MOSES A. BOUDOURIDES

In both society and nature, communities have always been ubiquitous as elementary forms of organization and have been also accepted intuitively as the niches and loci inside which humans (or possibly other living beings too) place and identify themselves according to various contextual, geographical, historical, cultural, political etc. conditions. Our aim here is to present an introductory and brief discussion of the formal concept of community in the context of the theory of complex networks (and social network analysis) and to describe (mostly by examples) a few of the many computational techniques which are commonly used for the detection of communities in a graph-theoretic background. This means that many interesting topics and methodologies in this field will have to be omitted, but fortunately there is a handful of widely available literature, which canvasses a panorama of almost the totality of the advances achieved up to now in this field. For instance, see the two excellent review articles of Fortunato (2010) and Porter *et al.* (2009).

Let us start with the formal background over which we will develop the discussion of communities. This is the notion of a simple¹ graph, which is constituted by a set of vertices (or nodes) together with a set of links (or connections) among them.² In any case, a graph might be directed (wherein links are identified as arcs) or undirected (links are called edges). Furthermore, a graph might be binary (also called dichotomous), in which case any link between two nodes either exists or does not exist. Or a graph might be weighted, in which case any link is equipped with a certain weight or value, which is, typically, a (positive) integer.

By a *community structure* of such a graph, we mean a partition of the set of nodes into a number of groups, called *communities*, such that all nodes belonging to any one of these groups satisfy a certain property of relative cohesiveness. Note that one may consider partitions, which are not necessarily strict, i.e., one may allow the case of *overlapping communities*, when there exist graph nodes belonging to more than one groups (communities) of the partition.

For instance, a typical property defining such a structure in a given graph is that communities should be relatively tight-knit, in the sense that nodes inside a community should be relatively densely connected to each other, while nodes among different communities should be relatively sparse. Of course, there are two extremes on the size of a community partition under such a condition of cohesiveness. On the lower end, every node becomes a community and, on the upper end, communities coincide with graph components. So, any nontrivial community partition ranges in

¹A graph is called simple when there exist neither multiple links nor self-loops.

²Without any loss of generality, we are using here the mathematical term *graph* as a synonym to *social network*. Of course, in sociology or other social sciences, nodes of a social network are interpreted as actors or agents and their links as (social) ties.

between these two extremes. Furthermore, one should recall that the idea of community partitioning based on cohesiveness was preceded by the structural notion of what in the sociological literature was called *cohesive groups*, i.e., “subsets of actors among whom there are relatively strong, direct, intense, frequent, or positive ties” (Wasserman & Faust, 1994, p. 249). It is exactly in this sense that communities in the context of structural sociology are perceived as groups of agents such that those within a community interact with each other more frequently (more intensely etc.) than with those outside the community.

However, in practice, operationalizing the gist of communities as cohesive groups is not a unanimously implemented task. In fact, there are many and diverse understandings of what communal cohesiveness means in formal graph-theoretic terms, which all correspond computationally to different techniques of community detection. Nevertheless, most of these methods of community partitioning share certain common features. For instance, communities are often seen as the *modules* of a rather complicated structure, which is often expected to exhibit the form of an *hierarchy*. Moreover, the process that derives communities as modules is assumed to operate in an iterative fashion until one of two terminal modular configurations is reached (i.e., until each module is either a singleton or a graph component). Thus, such a hierarchical partitioning can be represented as a tree, a *dendrogram*, which is composed of a hierarchy of nested modules.³ Of course, what one usually accepts as the community structure of a given graph is a set of modules obtained at some “reasonable” step⁴ of the iterative process and, in particular settings, there exist certain criteria construing what reasonable means (e.g., “optimal”).

At all events, strategies for building a modular hierarchical clustering generally fall into one of two types: divisive or agglomerative methods. On the one hand, a *divisive* clustering process is a “top down” approach, which starts initially from the (connected) graph considered as a single module (or, separately, from any component of a disconnected graph); and, as the divisive process develops, the root module(s) is (are) split recursively in more modules, while moving down the hierarchy, so that at each step modules are disassembled until they end up to becoming singletons (i.e., composed of single nodes). On the other hand, an *agglomerative* clustering process is a “bottom up” approach, which starts initially from a singleton module and it goes on merging together modules, while moving up the hierarchy, so that at each step bigger chunks of the graph are glued together until one reaches the whole graph (or its components).⁵

In the sequel, we have chosen to examine three particular methods, corresponding to three of the most popular methodologies of community detection. The first is a divisive approach, based on the well-known notion of betweenness-centrality. The second is a local-community finding approach, a variant of the agglomerative

³In such a dendrogram, the nodes of the graph are at the bottom (as leaves composed of singleton modules) and the components of the graph are on top (thus, when the graph is connected, the whole graph becomes the root module, while, otherwise, one observes a forest rooted on each component).

⁴Notice that, if an iterative clustering process gives rise to a dendrogram, then the step of the process becomes the depth of all modules produced at that step.

⁵In practice, due to certain computationally unsurpassable problems, merges and splits are usually determined in a *greedy* manner, i.e., following the problem solving metaheuristic of making the locally optimal choice at each step with the hope of eventually converging to the global optimum.

methodology, and it is based on the notion of k -cliques. The third is an optimization methodology, which is based on a particular quality function, called modularity.⁶ In each of these three methods, after first explaining the corresponding construal of group cohesiveness in the definition of communities, which is employed in that approach, we will briefly sketch (in an informal descriptive manner) how the method is technically operationalized and we will exhibit an illustrative example.⁷

1. Betweenness–Centrality–Based Community Detection

Let us first focus on a divisive method of hierarchical clustering for non-overlapping community partitioning, which was proposed by Michelle Girvan and Mark Newman (2002). This method is based on the notion of *betweenness centrality*, which was first elaborated in the context of social network analysis (Wasserman & Faust, 1994, pp. 189–191). In an undirected graph⁸, the betweenness of an edge is defined as the number of shortest (geodesic) paths between pairs of nodes that run through that edge. (Note that, in the classical definition of betweenness, one considers nodes instead of edges, but in exactly the same way.) Thus, in a sense, betweenness of edges can be construed as a measure of the (network) “traffic” of the “flow,” which circulates along the edges of the graph.

Now, the basic idea of this method is that two communities (seen as potentially tightly-knit cohesive groups) should be connected by (local) edges that operate as bridges, through which the circulating traffic (among communities) should be (relatively) high. Therefore, by ranking all edges according to their betweenness and then removing the edge(s) with the largest value, one could obtain the largest modules of this community partition at the first step. Recalculating betweenness for the remaining edges inside the first modules yields the communities of the second step. (Note that by removing some edges, the graph traffic is necessarily re-organized, since some previously low-traffic edges might then acquire higher traffic.) In this way, proceeding iteratively, one implements a divisive algorithm for detecting community structure in undirected graphs.

⁶Of course, there are many other methodologies for community detection than the ones we have chosen to discuss here and most of these methods are presented in the review articles of Fortunato (2010) and Porter *et al.* (2009). However, it might be proper just to mention the names of certain of these methodologies that we have left aside in the present brief and elementary introduction: spectral methods, methods based on short random walks, block-modeling, latent space clustering and statistical mechanics (such as Potts spin glasses).

⁷The first two examples were implemented computationally in the R package `iGraph`, through the functions displayed in the `IGraph::Community` module (<http://igraph.rubyforge.org/igraph/classes/IGraph/Community.html>), while the third example was worked out in the MATLAB computational environment, using the functions of the `Brain Connectivity Toolbox` (<https://sites.google.com/a/brain-connectivity-toolbox.net/bct/Home>).

⁸From now on, when we refer unqualifiedly to a graph, we mean a binary (or dichotomous) graph. Whenever we want to refer to a weighted graph, we will call it so explicitly.

As an example, let us give the social network of Zachary's karate club (1977), an undirected graph of 34 nodes and edges representing friendship ties among students at a university karate club, which is plotted below.

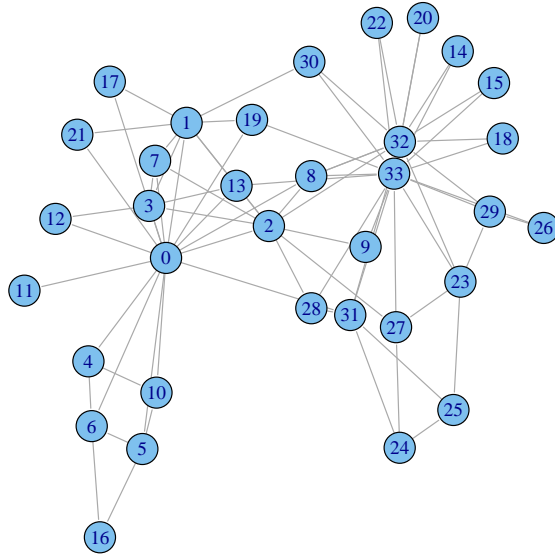


FIGURE 1. The Karate network.

Applying the Girvan–Newman algorithm of edge betweenness for the community detection of the karate network, we obtain the following dendrogram.

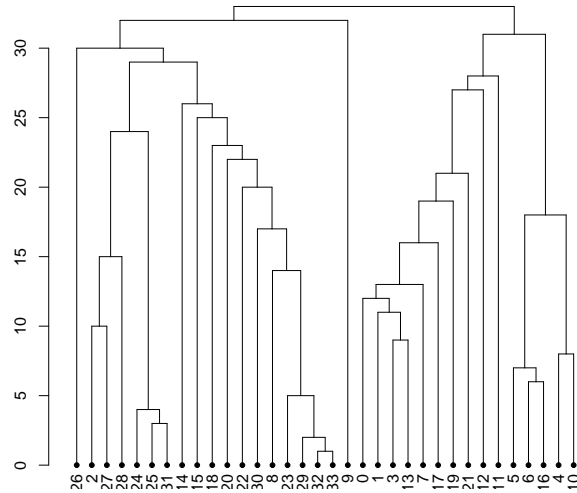


FIGURE 2. The dendrogram of the Karate network (by edge betweenness).

For instance, at (dendrogram) depth equal to 29, the following 5 communities of the karate network emerge.

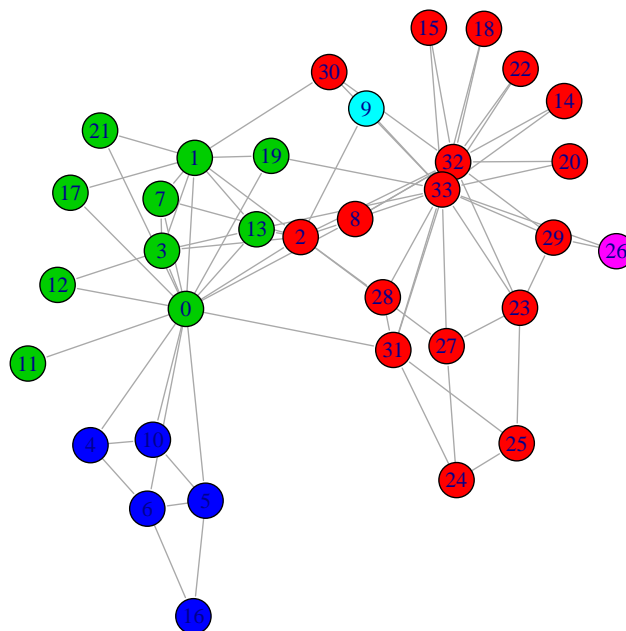


FIGURE 3. The 5 communities of the Karate network (by edge betweenness, at depth = 29).

2. k -Clique Percolation

The method of k -clique percolation is a community detection approach introduced by Palla *et al.* (2005) for analyzing the overlapping community structure of undirected or directed or weighted graphs (the extension to the latter two cases was due to Palla *et al.*, 2007 and Farkas *et al.*, 2007). The basic idea of this method is based on the intuitive answer to the following question: How could cliques be formed in a graph? Of course, one could imagine some sort of a process of transitive closure of edges that would be responsible for filling up with edges a part of the graph so that it might become a clique. But where could these edges come from? According to Palla *et al.*, the edges which are needed for the completion of a clique are more likely to come from the same component, where the process starts, due to the fact that internal edges inside a community are expected to have higher density than what trans-community edges have. In other words, if cliques were mobile and could move on a graph, the idea of this method is that cliques would not be able to travel everywhere, but most probably they would be trapped inside the community, from where they start moving, since it would be highly improbable for a clique to cross the bottleneck formed by the trans-community edges. Apparently, this method conceives communities not simply as relatively dense regions of the graph, but also as plastic structures, which are apt to carry through a process of local organization of the graph, a process that first launches locally and then percolates globally all over the graph.

To implement this idea, Palla *et al.* introduce the notion of a *k-clique* as a complete subgraph composed of k nodes. Note that this definition is different from the definition of an n -clique used in social network analysis, where an n -clique is considered to be a maximal subgraph, in which the largest geodesic distance between any two nodes is no greater than n (Wasserman & Faust, 1994, p. 258). Moreover, two k -cliques are said to be adjacent if they share $k - 1$ vertices, i.e., if they differ only at a single node. The union of adjacent k -cliques is called *k-clique chain*. Two k -cliques are connected if they are part of a k -clique chain. In this way, the k -clique percolation approach manages to formalize the simple observation that communities seem to be composed of several small cliques, which tend to share a number of both nodes and edges inside the same community. This is done by defining a *k-clique community* as the largest connected subgraph obtained by the union of a k -clique and of all k -cliques, which are connected to it. In other words, a k -clique community is defined as the union of k -cliques that can be reached from one to the other through a sequence of adjacent k -cliques.

In this way, a k -clique community is construed to be identified by allowing a *k-clique template* (an object isomorphic to a complete graph of k nodes) to “roll” over other adjacent k -cliques, where rolling means rotating a k -clique about the $k - 1$ nodes it shares with any adjacent k -clique. Therefore, defining a community through a potential process of percolation of such a template, a k -clique community is composed of the union of all subgraphs that can be fully probed by rolling a k -clique template. Obviously, when k increases, then it is harder to locate those communities, which would be able to facilitate the rolling of such a template. Therefore, the most appropriate value of k is 3 or slightly bigger (usually, it is taken up to 6, depending on the size and the density of the graph), given that the lower cases are trivial, as $k = 2$ reduces to bond (link) percolation and $k = 1$ to site (node) percolation.

From the above description of the method, it is clear that k -clique percolation belongs to the general category of *local community-finding* approaches. Although it is pertinent to the agglomerative approach, which also moves from the local to the global, it differs from it to the extent that (when $k \geq 2$) k -clique percolation fires up from elementary graph constituents (cliques), which are not reducible to single nodes (as the agglomerative approach does). Another difference with agglomerative (but also with divisive) techniques comes from the fact that, by construction, it is possible that k -clique communities may share nodes and so they can be overlapping. Furthermore, k -clique percolation may produce nodes that do not belong to any community at all (since they happen not to be parts of any k -clique in the graph).

As an example, we plot (in Figure 4 below) the Karate network, when communities are detected by the k -clique percolation method (for $k = 3$).

3. Modularity Maximization

Modularity maximization is one of the most widely used methods for community detection. Originally, it was introduced for undirected graphs (Newman, 2004), but it has subsequently extended to directed and even weighted graphs (Leicht & Newman, 2008). It is based on the notion of *modularity*, which is a benefit function that measures the quality of a particular partitioning of a graph into communities. The

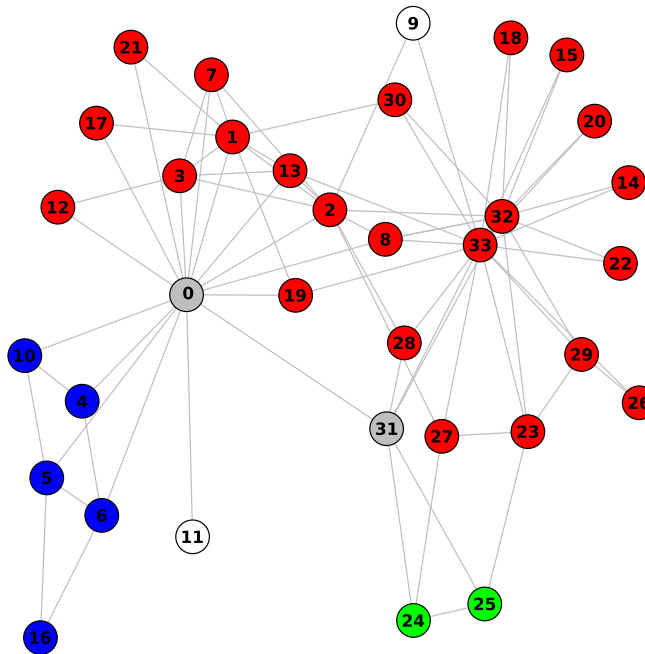


FIGURE 4. The communities of the Karate network (by k -clique percolation for $k = 3$): gray nodes are overlapping and white nodes do not belong to any community.

modularity maximization method detects communities by searching over possible partitions of a graph, over which modularity is maximized.

In its original definition (Newman & Girvan, 2004), modularity is based on the idea that, in a random graph, there should be no partitioning structure, which would be expected to be seen. Therefore, the appearance of any community partitioning in the given graph (usually derived from real data) would be revealed by the comparison between the actual density of edges in a community, considered as a subgraph, and the density one would expect to have in that subgraph, if the vertices of the graph were connected to each other regardless of any community partitioning. This expected edge density depends on the chosen *null model*, that is an almost random copy of the original graph, which does not exhibit any community structure, but, nonetheless, it possesses some of the structural properties of the original graph. Thus, the benefit function of modularity can be written as:

$$Q = \frac{1}{2m} \sum_{i,j} (A_{ij} - P_{ij}) \delta(C_i, C_j),$$

where the sum runs over all pairs of vertices of the graph, A is the adjacency matrix (i.e., $A_{ij} = 1$, whenever vertices i and j are connected, $A_{ij} = 0$, otherwise), m the total number of edges of the graph (note that, if k_i denotes the degree of i , then $\sum A_{ij} = \sum k_i = 2m$), P_{ij} represents the expected number of edges between vertices i and j in the null model, C_i denotes the community to which vertex i

belongs and the δ -function is defined as $\delta(C_i, C_j) = 1$, whenever i and j belong to the same community, and $\delta(C_i, C_j) = 0$, otherwise. Of course, the choice of the null model graph is in principle arbitrary and several other possibilities exist. For instance, one could assume that the null model graph keeps the same number of edges as the original graph and that edges are placed with the same probability between any pair of vertices. Then, the null model graph would become a Bernoulli random graph and the expected number of edges between vertices i and j would be constant (i.e., $P_{ij} = 2m/[n(n-1)]$, for all vertices i and j). However, such a null model would yield an inappropriate description of real networks, since its degrees follow the Poisson distribution, while, typically, in real networks, they should follow a skewed distribution (scale-free networks). For this reason, one prefers to use the null model having the same degree distribution with the original graph. In this model, the probability that vertices i and j are connected (on the condition that their degrees k_i and k_j are given) can be easily calculated. Actually, fixing vertex i , we would have a probability $p_i = k_i/2m$ that this vertex is connected to any other vertex and the corresponding probability for j would be $p_j = k_j/2m$. Since these are two independent events, the probability that vertices i and j are connected is $p_i p_j = k_i k_j / 4m^2$ and the expected number of edges between i and j is $P_{ij} = 2mp_i p_j = k_i k_j / 2m$. Thus, the expression of modularity becomes:

$$Q = \frac{1}{2m} \sum_{i,j} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(C_i, C_j).$$

Therefore, as the only contributions to the above sum come from pairs of vertices belonging to the same community, we obtain:

$$Q = \sum_{k=1}^{n_c} \left[\frac{l_k}{m} - \left(\frac{d_k}{2m} \right)^2 \right],$$

where n_c is the total number of communities in a partition, l_k is the total number of edges inside community C_k and d_k is the sum of degrees of all vertices in C_k . In other words, each summand of the above sum (corresponding to a community) is computed as the difference between the fraction of edges of the graph inside a community minus the expected fraction of edges that would be there if the graph were a random graph with the same degrees of vertices.⁹

Therefore, good partitions should give large positive values of Q . In fact, the way modularity was defined (as it has been normalized), Q ranges between -1 and $+1$. $Q = 0$ means that all vertices of the graph are assigned to the same community (i.e., $n_c = 1$, for the trivial partition in one community consisting of the whole graph). On the other extreme side, if we have a partition with each community being a singleton (i.e., $n_c = n$, where n denotes the number of vertices of the graph), then Q turns out to be negative. In particular, if for any partition of a graph, modularity is found non-positive, then this graph has no community structure (in fact, such a graph would exhibit a strong multipartite structure, in the sense that it would be decomposed to certain subgraphs with very few internal edges and many edges lying between them). In any case, if, for a partition, modularity happens to take large positive values (close to 1), then this indicates that the graph is decomposed to communities and most of the edges of the graph fall within these communities

⁹Note that the above definition of modularity is strongly related to Lin Freeman's *segregation index* (Freeman, 1978).

than what would have been expected by chance (under the previously discussed null model).

Although the definition of modularity is intuitively straightforward, if one wants to implement it computationally in a procedure of an exhaustive optimization, which would run through all possible partitions that can be formed in a given graph, then it becomes, alas, an impossible task. As a matter of fact, it has been proven to be an intractable NP-complete problem (Brandes *et al.*, 2008). Consequently, many practical algorithms were devised for the graph modularity computation, offering different balances between speed and accuracy. Most of these methods are based on approximate optimization techniques, such as greedy algorithms, simulated annealing, extremal optimization, spectral algorithms etc. (discussed with references in the review article of Fortunato, 2010).

Finally, as an example, we give a directed and weighted graph, which was formed from the messages exchanged in the mailing list *Multilevel* during May 1999. Here, an arc connects the *sender* of a message distributed in the list with the exact *recipient*, to whom the content of the message was addressed by providing answers or posing questions or expressing comments, remarks etc.¹⁰ The following is a plot of this graph of the mailing list *Multilevel*.

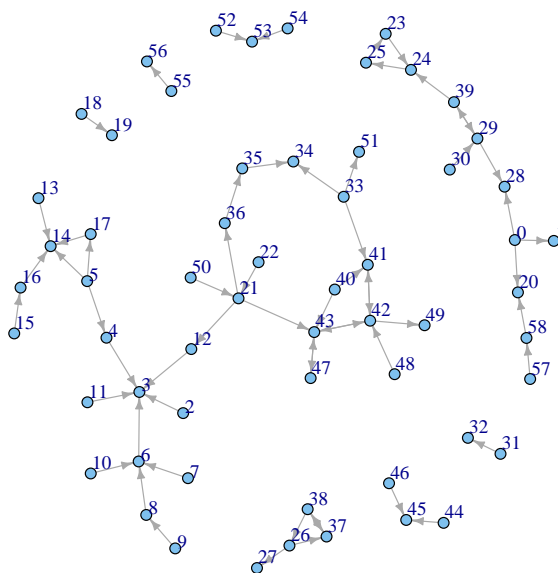


FIGURE 5. The directed (weighted) graph of the mailing list *Multilevel* in May 1999.

Applying the Leicht–Newman modularity maximization technique, we have found 12 communities in the optimal community structure. This is a partition of the graph

¹⁰The data of this directed graph were mined over the list archives by a script implementing certain rules for the search of the recipients, which was written by Yiorgos Adamopoulos, from the University of Piraeus, Greece.

into non-overlapping communities such that the number of within-communities arcs is maximized, while the number of between-communities arcs is minimized. However, note that this optimal number of communities might vary in multiple runs of the algorithm, as Good *et al.* (2010) argue in their study of performance of modularity maximization in practical contexts.

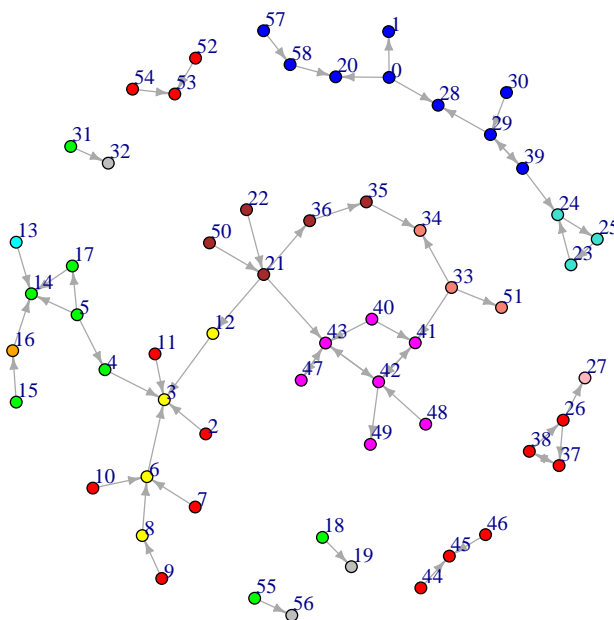


FIGURE 6. The 12 communities of the graph of the mailing list *Multilevel* in May 1999 (detected by the method of modularity maximization of Leicht Newman, 2008).

REFERENCES

- Brandes, U., Delling, D., Gaertler M., Goerke, R., Hofer, M., Nikoloski, Z., & Wagner, D. 2008. On modularity clustering. *IEEE Transactions on Knowledge and Data Engineering*, **20**, 172–188. <http://dx.doi.org/10.1109/TKDE.2007.190689>.
- Farkas, I., Palla, G., & Vicsek, T. 2007. Weighted network modules. *New J. Phys.*, **9**, 180. <http://dx.doi.org/10.1088/1367-2630/9/6/180>.
- Fortunato, S. 2010. Community detection in graphs. *Phys. Rep.*, **486**, 75–174. <http://dx.doi.org/j.physrep.2009.11.002>.
- Freeman, L.C. 1978. Segregation in social networks. *Sociological Methods Research*, **6**, 411–429. <http://dx.doi.org/10.1177/004912417800600401>.
- Girvan, M., & Newman, M.E.J. 2002. Community structure in social and biological networks. *Proc. Natl. Acad. Sci. USA*, **99**, 7821–7826. <http://dx.doi.org/10.1073/pnas.122653799>.

- Good, B.H., de Montjoye, & Clauset, A. 2010. The performance of modularity maximization in practical contexts. *Phys. Rev. E*, **81**, 046106. <http://dx.doi.org/10.1103/PhysRevE.81.046106>.
- Leicht, E.A., & Newman, M.E.J. 2008. Community structure in directed networks. *Phys. Rev. Lett.*, **100**, 118703. <http://link.aps.org/doi/10.1103/PhysRevLett.100.118703>.
- Newman, M.E.J. 2004. Fast algorithm for detecting community structure in networks. *Phys. Rev. E*, **69**, 066133. <http://dx.doi.org/10.1103/PhysRevE.69.066133>.
- Newman, M.E.J., & Girvan, M. 2004. Finding and evaluating community structure in networks. *Phys. Rev. E*, **69**, 026113. <http://dx.doi.org/10.1103/PhysRevE.69.026113>.
- Palla, G., Derényi, I., Farkas, I., & Vicsek, T. 2005. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, **435**, 814–818. <http://dx.doi.org/10.1038/nature03607>.
- Palla, G., Farkas, I.J., Pollner, P., Derényi, I., & Vicsek, T. 2007. Directed network modules. *New J. Phys.*, **9**, 186. <http://dx.doi.org/10.1088/1367-2630/9/6/186>.
- Porter, M.A., Onnela, J.-P., & Mucha, P.J. 2009. Communities in networks. *Notices Amer. Math. Soc.*, **56**, 1082–1097, 1164–1166. <http://www.ams.org/notices/200909/rtx090901082p.pdf>.
- Wasserman, S., & Faust, K. 1994. *Social Network Analysis: Methods and Applications*. 1st edn. Cambridge: Cambridge University Press.
- Zachary, W.W. 1977. An information flow model for conflict and fission in small groups. *J. Anthropological Res.*, **33**, 452–473.