Les Cahiers du GERAD ISSN: 0711–2440

A new centrality-based approach for fast community detection in graphs

E. Camby
G. Caporossi

G–2016–33

May 2016

Cette version est mise à votre disposition conformément à la politique de libre accès aux publications des organismes subventionnaires canadiens et québécois.

Avant de citer ce rapport, veuillez visiter notre site Web (https://www.gerad.ca/fr/papers/G-2016-33) afin de mettre à jour vos données de référence, s'il a été publié dans une revue scientifique.

This version is available to you under the open access policy of Canadian and Quebec funding agencies.

Before citing this report, please visit our website (https://www.gerad.ca/en/papers/G-2016-33) to update your reference data, if it has been published in a scientific journal.

Les textes publiés dans la série des rapports de recherche *Les Cahiers du GERAD* n'engagent que la responsabilité de leurs auteurs.

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2016
– Bibliothèque et Archives Canada, 2016

The authors are exclusively responsible for the content of their research papers published in the series *Les Cahiers du GERAD*.

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2016
– Library and Archives Canada, 2016

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

A new centrality-based approach for fast community detection in graphs

Eglantine Camby [a]

Gilles Caporossi [b]

[a] *Département de Mathématique, Université Libre de Bruxelles, Bruxelles, Belgique*

[b] *GERAD & HEC Montréal, Montréal (Québec) Canada*

Eglantine.Camby@ulb.ac.be
gilles.caporossi@gerad.ca

May 2016

Les Cahiers du GERAD

G–2016–33

Copyright © 2016 GERAD

Abstract: In this paper, we propose a new scheme for building algorithms to detect communities in networks. This new approach is based upon a vertex centrality measure and the hypothesis that central vertices, i.e. vertices with high centrality, link distinct communities. It turns out that the betweenness centrality and the centrality based upon the clustering coefficient provide good results, but it is also the case for the eigenvector centrality. These methods are experimented on 11 classical networks and results are compared to those produced by Blondel's algorithm [6], polynomial-time heuristic optimizing the modularity centrality, and those by an algorithm [2] finding the optimal partition for the modularity centrality. Based on this new approach, the resulting algorithm runs, beyond the centrality computation, in $O(\max(n\log(n), m))$. These preliminary results are encouraging and will lead to further research for finding appropriate centrality measures and study their performance.

Key Words: Complex networks, community detection, centrality analysis, algorithms.

Acknowledgments: This work was partially supported by a post-doc grant "Bourse d'Excellence WBI.WORLD" from Fédération Wallonie-Bruxelles (Belgium) as well as by NSERC (Canada).

Introduction

When searching for communities in networks, the first question that arises is to know the definition of a community. Unfortunately, there is no exact definition of communities, but it is generally accepted that communities are groups of individuals that are more related to each others than to the individuals from other communities. From an intuitive way, individuals from the same community are related, which means that they more or less know each others.

A measure of belonging to a community is the clustering coefficient [29] which is the proportion of neighbors of a vertex that are adjacent. The value of the clustering coefficient lies between 0 and 1. It is 1 when all the neighbors of the vertex are adjacent (which means that the vertex and its neighbors make a clique), and it is 0 when none of the neighbors are adjacent (which tends to indicate that the vertex is actually a bridge between communities). Observe that a centrality measure can be obtained from the clustering coefficient by subtracting his value from 1. It is certainly not by chance if the clustering coefficient was used as a starting point of the algorithm from Castellano, Cecconi, Loreto, Parisi, and Radicchi [28] for community detection.

It is also not surprising to see that the betweenness centrality [17, 18] was also used for that same purpose. Indeed, the betweenness centrality can be also computed for each vertex and is defined as the number of shortest paths between pairs of vertices and containing this vertex. When studying these two measures and their bounds with AutoGraphiX [15, 14], we notice that the upper bound of the clustering coefficient is achieved for a vertex when and only when the lower bound of the betweenness centrality is too for the same vertex, and actually this vertex has all its neighbors pairwise adjacent.

A contribution of this paper is the following approach: communities are not built around influential individuals, i.e. vertices with high centrality, but at the opposite, influential individuals actually link communities. From now, these vertices are called central, due to their role. Therefore, in terms of clustering coefficient, influential individuals correspond to vertices with a high centrality, i.e. a low clustering coefficient, approaching to 0.

Accordingly, the problem of finding communities may be handled from a different point of view. Community detection may be achieved by identifying a set of vertices that make links between communities. Based on the above mentioned remarks, we propose an algorithm in two steps to identify kernels of communities. The first step is to find central vertices, corresponding to influential individuals. After removing central vertices, each remaining connected component is then the kernel of a community.

The algorithm proposed by Newman and Girvan [27] was based upon a related principle. Indeed, they used the betweenness centrality as a measure to find edges with a specific role. According to them, edges with a high betweenness are links between different communities. Therefore, their algorithm removes iteratively these edges and, after each removal, the betweenness centrality is recalculated. The communities then appear as connected components in the remaining graph.

The main differences in the present paper compared to [27] are that our approach is valid for any centrality measures and we suggest to remove vertices instead of edges. This approach seems more promising in the case some vertex belongs to different communities and is strongly connected to each of them. In such a case, no edge has a significant weight, which prevents the separation of the communities. Since each vertex has a local influence, instead of sorting the vertices according to its centrality over the whole graph, we propose to compare its centrality to its neighbors. It is another difference.

The new algorithm aims at finding communities in graphs using different vertex-related measures. It turns out that the betweenness centrality or the local clustering coefficient may both be used as a starting point, but each of them has some limitations. First, the clustering coefficient can not be used in the case of bipartite graphs, which is problematic since there are a wide variety of situations that may be modeled by bipartite graphs. Second, the betweenness centrality could be applied in any case, but it implies to compute shortest paths between pairs of vertices. If this computation is polynomial, it is difficult to achieve in very large graphs in a reasonable time. For these reasons, we propose also the use of eigenvector centrality [10, 12]

as an alternative. Using the power iteration algorithm, this centrality may be computed very efficiently even for very large graphs.

The description of the algorithm is given in the first section. In the second section, some numerical results are provided and the obtained partitions are compared to those obtained by modularity [27] maximization (exact algorithm [2] and heuristic with the Blondel's algorithm [6]). Finally, in a third section, some issues are raised and discussed in the conclusion.

1 Description of the algorithm

Based on the principle that central vertices make the link between communities, the first step of the algorithm is to identify these vertices, according to a centrality measure. Various centrality measures may be used, among which the most promising are the betweenness centrality [17, 18] and the centrality deduced from the clustering coefficient [29]. Other centrality measures may also be considered, such as the eigenvector centrality [10, 11] for instance.

It may occur that the vertices with higher centrality are close to each other. Moreover, since linking various communities is a local property, instead of considering vertices with higher centrality over the whole graph, we consider vertices with higher centrality relative to its neighborhood. Formally, central vertices are vertices with a centrality greater than a threshold depending on its neighborhood. In this paper, we investigate when the threshold of a vertex equals the average of the centrality measure of its neighbors.

Require: A connected graph G
Ensure: A partition of the vertices in communities
$S \leftarrow \emptyset$;
Compute c_v the centrality of each vertex $v \in V$;
Order the vertices with increasing centrality c_v, breaking ties arbitrarily;
Compute $t(v)$ the threshold above which v is considered *central*, i.e. $t(v) \leftarrow \frac{\sum_{u \in N(v)} c_u}{|N(v)|}$;
if $c_v > t(v)$ **then**
  $S \leftarrow S \cup \{v\}$;
end if
Let $G' = G \setminus S$ be the graph obtained from G by removing vertices of S;
Let $\mathscr{P}$ be the partition of G' in connected components (communities);
while $S \neq \emptyset$ **do**
  for each $v \in S$ (by increasing order of c_v) **do**
    Assign v to a community in which it has the greatest number of neighbors;
    $S \leftarrow S \setminus \{v\}$;
  end for
end while
for each v that is alone in its community **do**
  Assign v to a community in which it has the greatest number of neighbors;
end for return $\mathscr{P}$;

Algorithm 1: Centrality-Clustering Algorithm.

The so-identified central vertices are then removed from the graph, most likely leading the remaining graph to be disconnected. Each connected component of the graph is then considered as the kernel of a community. Some post-treatments are then applied. They consist in assigning central vertices to communities, and reassigning vertices that are alone in their community. In the first post-treatment, considering central vertices by increasing order of centrality is really important to ensure that each vertex has at least one neighbor which is non-central.

2 Numerical results

In order to analyze the results obtained by Centrality-Clustering Algorithm, called CC Algorithm, as a first step, we restrict our experimentation of finding communities to 11 classical networks presented in [2] since they are the networks for which the optimal modularity solution is known. A set of 10 larger networks from the literature is then added in order to study the computational performance of the algorithm.

Table 1 compares the results obtained by CC Algorithm according to three centrality measures (eigenvector centrality, betweenness centrality and that deduced from the clustering coefficient) with the best (according to the modularity) solution among 100 runs of Blondel's algorithm [6] and the optimal modularity solution. We investigate results given by the Blondel's algorithm, often used in practice, since this polynomial-time algorithm is a heuristic to optimize the modularity, but also the optimal modularity solution, known in these networks by the exact algorithm from Aloise, Cafieri, Caporossi, Hansen, Perron and Liberti [2], in order to avoid bias induced by heuristic.

The criteria of comparison are based on the number M of communities in the partition, the average density $\overline{den}$, the edge-ratio ER, the coverage $covr$, the performance per and the modularity Q, all defined by equalities (1)–(5).

$$\overline{den} = \frac{1}{M}\sum_{k=1}^{M} den_k \tag{1}$$

$$ER = \frac{m_{in}}{m_{out}} \tag{2}$$

$$covr = \frac{m_{in}}{m} \tag{3}$$

$$per = \frac{2(m_{in} + pairs_{out} - m_{out})}{n(n-1)} \tag{4}$$

$$Q = \sum_{k=1}^{M} \frac{m_{in}^k}{m} - \left(\frac{2m_{in}^k + m_{out}^k}{2m}\right)^2 \tag{5}$$

where n and m denote, as usual, the number of vertices and edges, den_k is the density (i.e. the ratio between the number of edges and the number of all possible edges) of the k^{th} community, m_{in} is the number of edges whose ends are in the same community, m_{out} is the number of edges joining vertices from different communities (i.e. $m_{out} = m - m_{in}$), $pairs_{out}$ is the number of pairs of vertices from different communities, while m_{in}^k, respectively m_{out}^k, represents the number of edges with both ends in the community k, respectively the number of edges with exactly one end in the community k.

The coverage, the performance and the modularity are three criteria used by Fortunato [16] while the average density and the edge-ratio are natural criteria. However, every criterion has a weakness, i.e. there exist a network and a partition which is not properly evaluated according to this criterion.

Table 1: Results for the main classical benchmark datasets and their comparison to partitions obtained by modularity maximization (using Blondel's algorithm and exact aglorithm).

Algorithm	M	avg. density	edge-ratio	coverage	performance	modularity
Karate [31]						
CC- Eigenvector	4	0.56	3.33	0.77	0.75	0.402
CC- Betweenness	3	0.40	2.71	0.73	0.64	0.274
CC- Cluscoef	4	0.56	3.33	0.77	0.75	0.402
Blondel Modularity	3	0.35	3.11	0.76	0.71	0.381
Optimal Modularity	4	0.45	2.71	0.73	0.80	0.420
Dolphins [24]						
CC- Eigenvector	1	-	-	-	-	-
CC- Betweenness	3	0.47	13.45	0.93	0.52	0.330
CC- Cluscoef	9	0.65	1.79	0.64	0.85	0.426
Blondel Modularity	4	0.42	4.68	0.82	0.74	0.495
Optimal Modularity	5	0.36	3.18	0.76	0.82	0.529

Continued on next page ...

... *Continued from previous page*

Algorithm	M	avg. density	edge-ratio	coverage	performance	modularity
les Misérables [21]						
CC- Eigenvector	7	0.72	11.10	0.92	0.60	0.203
CC- Betweenness	5	0.67	14.87	0.94	0.54	0.260
CC- Cluscoef	11	0.77	3.16	0.76	0.90	0.527
Blondel Modularity	5	0.46	2.74	0.73	0.82	0.501
Optimal Modularity	6	0.42	3.23	0.76	0.88	0.560
A00 main [4]						
CC- Eigenvector	3	0.28	5.94	0.86	0.37	0.182
CC- Betweenness	11	0.47	3.81	0.79	0.84	0.474
CC- Cluscoef	15	0.49	1.36	0.58	0.92	0.417
Blondel Modularity	8	0.28	3.17	0.76	0.87	0.527
Optimal Modularity	9	0.27	2.57	0.72	0.90	0.531
p53 [22]						
CC- Eigenvector	3	0.68	31.29	0.97	0.12	0.017
CC- Betweenness	6	0.49	7.37	0.88	0.43	0.180
CC- Cluscoef	13	0.70	2.14	0.68	0.76	0.370
Blondel Modularity	8	0.30	2.37	0.70	0.86	0.523
Optimal Modularity	7	0.28	2.65	0.73	0.85	0.535
Polbooks [3]						
CC- Eigenvector	3	0.45	8.19	0.89	0.37	0.131
CC- Betweenness	4	0.46	10.61	0.91	0.63	0.452
CC- Cluscoef	9	0.67	2.47	0.71	0.72	0.338
Blondel Modularity	4	0.44	11.25	0.92	0.69	0.502
Optimal Modularity	5	0.45	8.00	0.89	0.76	0.527
Football [19]						
CC- Eigenvector	2	0.55	33.06	0.97	0.12	0.003
CC- Btw/Cluscoeff	1	-	-	-	-	-
Blondel Modularity	7	0.60	2.88	0.74	0.88	0.577
Optimal Modularity	10	0.76	2.42	0.71	0.94	0.605
A01 main [4]						
CC- Eigenvector	5	0.74	34.28	0.97	0.09	0.015
CC- Betweenness	13	0.67	11.45	0.92	0.43	0.227
CC- Cluscoef	28	0.74	2.61	0.72	0.81	0.439
Blondel Modularity	11	0.24	3.74	0.79	0.88	0.613
Optimal Modularity	14	0.29	2.94	0.75	0.92	0.633
USAir97 [5]						
CC- Eigenvector	10	0.87	49.62	0.98	0.15	0.009
CC- Betweenness	19	0.77	26.97	0.96	0.41	0.067
CC- Cluscoef	29	0.83	11.01	0.92	0.52	0.101
Blondel Modularity	7	0.44	3.34	0.77	0.68	0.321
Optimal Modularity	6	0.14	1.92	0.66	0.81	0.368
netscience main [26]						
CC- Eigenvector	46	0.75	4.02	0.80	0.95	0.732
CC- Betweenness	53	0.73	4.47	0.82	0.97	0.764
CC- Cluscoef	76	0.81	2.36	0.70	0.99	0.673
Blondel Modularity	19	0.41	14.49	0.94	0.93	0.839
Optimal Modularity	19	0.35	11.35	0.92	0.95	0.849
s838 [25]						
CC- Eigenvector	22	0.34	6.58	0.87	0.66	0.574
CC- Betweenness	66	0.52	2.55	0.72	0.98	0.690
CC- Cluscoef	52	0.63	2.90	0.74	0.72	0.446
Blondel Modularity	13	0.07	8.30	0.89	0.93	0.809
Optimal Modularity	12	0.07	9.37	0.90	0.92	0.819

Due to the definition of the betweenness centrality and that deduced from the clustering coefficient, one would expect that CC Algorithm using those centralities would perform better than CC Algorithm based the upon eigenvector centrality. Surprisingly, this intuition is not verified since results based on the three

centralities are competitive, even with the results obtained from the modularity. Overall, none of these algorithms stands out for being the best or the worst. The hypothesis that a higher centrality is associated with vertices that lie between communities therefore seems realistic in practice.

Roughly, we note that, based on these networks, none algorithm gives a partition which is the best over all criteria. Even if Blondel's algorithm is designed to optimize the modularity, which is not the case of CC Algorithm, the modularity of partitions found by CC Algorithm can be greater than the modularity of those by Blondel's Algorithm, as it is the case in Karate and les Misérables networks. For a deeper analysis, we make a ranking table of the results obtained from algorithms with comparable complexity time. Therefore, we do not consider the partitions which are optimal for the modularity criterion. Table 2 describes the fractional rank of each polynomial-time algorithm over the 11 given networks, depending on each criterion. The fractional rank is computed as the average of ranks over 11 networks, knowing that the best rank is chosen in case of ties.

Table 2: Average of ranks of each algorithm over the 11 given networks.

Algorithm	average density	edge-ratio	coverage	performance	modularity
CC- Eigenvector	2.09	1.81	1.81	3.36	3.36
CC- Betweenness	2.72	2.27	2.27	2.90	2.72
CC- Cluscoef	1.27	3.27	3.27	1.63	2.45
Blondel Modularity	3.54	2.45	2.45	1.90	1.27

In view of Table 2, CC Algorithm with the centrality based upon the clustering coefficient seems to be better in terms of average density criterion, as well as CC Algorithm with the eigenvector centrality over the edge-ratio and the coverage criteria. Moreover, CC Algorithm based on the betweenness centrality remains stable in the ranking table for every criterion. Notice that as expected, Blondel's algorithm performs overall better on the modularity criterion. In terms of performance criterion, CC algorithm running with the clustering coefficient seems the best, slightly better than Blondel's algorithm.

One of the most important strengths of CC Algorithm is clearly its speed. CC Algorithm runs in $\mathcal{O}(\max(f(n,m), n\log(n), m))$, where n is the number of vertices, m is the number of edges and $f(n,m)$ is the computational complexity of the used centrality measure. Indeed, the computation of the centrality in Line 2 runs clearly in $\mathcal{O}(f(n,m))$. It is well-known that sorting vertices in Line 3 can be computed in $\mathcal{O}(n\log(n))$. In Line 4, the computation of threshold is done in $\mathcal{O}(m)$, while the running time of finding central vertices is $\mathcal{O}(n)$ in Lines 5–7, as well as that of removing central vertices (Line 8) is $\mathcal{O}(n)$. In Line 9, searching connected components in the resulting graph takes $\mathcal{O}(m)$. The two post-treatments (Lines 10–18) can be done in $\mathcal{O}(m)$.

Table 3 provides the time performance of CC Algorithm on 21 classical networks from the literature. Given that the most time consuming part of the algorithm is the centrality computation, the fastest of experimented centralities, i.e. the eigenvector centrality, was used to evaluate the computational time of CC Algorithm. The first column denotes the network, the two following ones (n and m) provide respectively the number of vertices and edges, the next two columns describe the performance of CC Algorithm. Indeed, the number M of communities is provided as well as the computational time in seconds, including the required time to build the network from its edge list description in a file. These experiments were achieved on a computer with 2 processors Intel Xeon-8 cores running at 2.4 GHz with 128 GB of RAM. Note that parallel capability of the computer was not used, as all the tests were achieved on a single thread.

3 Discussion and conclusion

3.1 Variations around CC Algorithm

The proposed algorithm is a framework that may be adapted by using different centrality measures, but also, by changing the threshold when selecting central vertices.

Table 3: Computational time of CC Algorithm using the eigenvector centrality.

Problem	n	m	M	Time (sec)
Karate [31]	34	78	4	0
Dolphins [24]	62	159	1	0
Les Misérables [21]	77	255	7	0
A00 main [4]	83	125	3	0
p53 [22]	104	226	3	0
Polbooks [3]	105	441	3	0
Football [19]	115	613	2	0
A01 main [4]	249	635	5	0
USAir97 [5]	332	2 126	10	0
netscience main [26]	379	914	46	0
s838 [25]	512	819	22	0.02
email [20]	1133	5451	5	0.03
polblogs main [1]	1 222	16 714	8	0.06
power [29]	4 941	6 594	4	0.05
erdos-02 [5]	6 927	11 850	37	0.06
PGP giant component [7]	10 680	24 316	220	0.19
cnr-2000 [8, 9]	325 557	2 738 969	4	45
eu-2005 [8, 9]	862 664	16 138 468	233	295
in-2004 main [8, 9]	1 353 703	13 126 172	683	71
as-skitter main [23]	1 694 616	11 094 209	585	287
friendster [30]	65 608 366	1 806 067 135	5116	22 770

Thus, an extension of CC Algorithm could be to change the threshold function $t(v)$. For instance, the maximum of the centrality of its neighborhood could be used :

$$t(v) = \max\{c_w | w \in N(v)\}, \tag{6}$$

as well as any convex combination of the latter and the used threshold, i.e.

$$t(v) = (1-\lambda)\max\{c_w | w \in N(v)\} + \lambda \frac{\sum_{u \in N(v)} c_u}{|N(v)|} \tag{7}$$

where λ is a real value between 0 and 1. If $\lambda > 1$, no vertex is considered as central and only one community could be found. At the opposite, a negative value of λ would imply that the threshold would be below the average, in which case the algorithm may not find a partition (for instance when all vertices have the same centrality, all vertices would then be considered as central).

Another way to adapt the algorithm is by using it recursively in order to find smaller communities. Each community found at iteration k is thus considered as a network in which communities will be found for the iteration $k+1$. The algorithm is then a divisive algorithm similarly to those proposed by Newman and Girvan [27] or Cafieri et al. [13]. Since CC Algorithm sometimes finds only one community (which occurs for the Dolphins and Football networks), the recursive version of CC Algorithm does not require any stopping criterion and will naturally stop dividing communities. This property avoids the question of the stopping criterion and ensures that the algorithm identifies the proper number of communities by itself. We expect that the communities found by the recursive version of CC Algorithm will likely be smaller and more homogeneous than those by CC Algorithm.

3.2 Concluding remarks

In this paper, we developed an approach for community detection in graphs. This approach is based on a centrality measure. We tested the betweenness centrality, a centrality based upon the clustering coefficient and the eigenvector centrality. To our knowledge, this approach is original since it is based upon a different paradigm than the density of clusters or edge ratio. Indeed, we consider that central vertices tends to link different communities. This intuition holds in practice for all the three tested centrality measures, by experimentation on 21 classical networks.

Using this property leads to Centrality-Clustering Algorithm. The algorithm is fast and does not require the number of communities to be provided. Extensions of the algorithm may still be achieved, especially toward the definition of new meaningful centrality definitions.

References

[1] L.A. Adamic and N. Glance. The political blogosphere and the 2004 us election: divided they blog. In Proceedings of the 3rd international workshop on Link discovery, pages 36–43. ACM, 2005.

[2] D. Aloise, S. Cafieri, G. Caporossi, P. Hansen, S. Perron, and L. Liberti. Column generation algorithms for exact modularity maximization in networks. Physical Review E, 82(4):046112, 2010.

[3] Anonyme. Books about us politics. http://networkdata.ics.uci.edu/data.php?d=polbooks, 2016. Accessed: 2016-05-17.

[4] Anonyme. Networks/pajekpajek. http://vlado.fmf.uni-lj.si/pub/networks/data/GD/GD.htm., 2016. Accessed: 2016-05-17.

[5] Anonyme. Pajek datasets. http://vlado.fmf.uni-lj.si/pub/networks/data/, 2016. Accessed: 2016-05-17.

[6] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E Lefebvre. Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment, 2008(10):P10008, 2008.

[7] M. Boguñá, R. Pastor-Satorras, A. Díaz-Guilera, and A. Arenas. Models of social networks based on social distance attachment. Physical review E, 70(5):056122, 2004.

[8] P. Boldi, M. Rosa, M. Santini, and S. Vigna. Layered label propagation: A multiresolution coordinate-free ordering for compressing social networks. In Sadagopan Srinivasan, Krithi Ramamritham, Arun Kumar, M. P. Ravindra, Elisa Bertino, and Ravi Kumar, editors, Proceedings of the 20th international conference on World Wide Web, pages 587–596. ACM Press, 2011.

[9] P. Boldi and S. Vigna. The WebGraph framework I: Compression techniques. In Proc. of the Thirteenth International World Wide Web Conference (WWW 2004), pages 595–601, Manhattan, USA, 2004. ACM Press.

[10] P. Bonacich. Factoring and weighting approaches to status scores and clique identification. Journal of Mathematical Sociology, 2(1):113–120, 1972.

[11] P. Bonacich. Power and centrality: A family of measures. American Journal of Sociology, pages 1170–1182, 1987.

[12] P. Bonacich. Some unique properties of eigenvector centrality. Social Networks, 29(4):555–564, 2007.

[13] S. Cafieri, P. Hansen, and L. Liberti. Locally optimal heuristic for modularity maximization of networks. Physical Review E, 83(5):056105, 2011.

[14] G. Caporossi. Variable neighborhood search for extremal vertices: The system autographix-iii. Technical Report G-2015-09, Groupe d'études et de Recherche en Analyse des Décisions, 2015.

[15] G. Caporossi and P. Hansen. Variable neighborhood search for extremal graphs: 1 the autographix system. Discrete Mathematics, 212(1):29–44, 2000.

[16] S. Fortunato. Community detection in graphs. Physics Reports, 486(3):75–174, 2010.

[17] L. C. Freeman. A set of measures of centrality based on betweenness. Sociometry, pages 35–41, 1977.

[18] L. C. Freeman. Centrality in social networks conceptual clarification. Social Networks, 1(3):215–239, 1979.

[19] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. Proceedings of the national academy of sciences, 99(12):7821–7826, 2002.

[20] R. Guimera, L. Danon, A. Diaz-Guilera, F. Giralt, and A. Arenas. Self-similar community structure in a network of human interactions. Physical review E, 68(6):065103, 2003.

[21] B. Karrer, El. Levina, and M. E. J. Newman. Robustness of community structure in networks. Physical Review E, 77:046119, 2008.

[22] K. W. Kohn. Molecular interaction map of the mammalian cell cycle control and dna repair systems. Molecular biology of the cell, 10(8):2703–2734, 1999.

[23] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, pages 177–187. ACM, 2005.

[24] D. Lusseau, K. Schneider, O. J. Boisseau, P. Haase, E. Slooten, and S. M. Dawson. The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations. Behavioral Ecology and Sociobiology, 54(4):396–405, 2003.

[25] R. Milo, S. Itzkovitz, N. Kashtan, R. Levitt, S. Shen-Orr, I. Ayzenshtat, M. Sheffer, and U. Alon. Superfamilies of evolved and designed networks. Science, 303(5663):1538–1542, 2004.

[26] M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. Physical Review E, 74(3):036104, 2006.

[27] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. Physical Review E, 69(2):026113, 2004.

[28] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi. Defining and identifying communities in networks. Proceedings of the National Academy of Sciences of the United States of America, 101(9):2658–2663, 2004.

[29] D. J. Watts and S. H. Strogatz. Collective dynamics of ?small-world? networks. Nature, 393(6684):440–442, 1998.

[30] J. Yang and J. Leskovec. Defining and evaluating network communities based on ground-truth. Knowledge and Information Systems, 42(1):181–213, 2015.

[31] W. W. Zachary. An information flow model for conflict and fission in small groups. Journal of anthropological research, pages 452–473, 1977.