



Identifying Well-Connected Communities in Real-World and Synthetic Networks

Minhyuk Park¹, Yasamin Tabatabaee¹, Vikram Ramavarapu¹, Baqiao Liu¹,
Vidya Kamath Pailodi¹, Rajiv Ramachandran¹, Dmitriy Korobskiy²,
Fabio Ayres³, George Chacko¹(✉), and Tandy Warnow¹(✉)

¹ University of Illinois Urbana-Champaign, Urbana, IL, USA

{chackoge,warnow}@illinois.edu

² NTT DATA, McLean, VA, USA

³ Insper Institute, São Paulo, Brazil

Abstract. Integral to the problem of detecting communities through graph clustering is the expectation that they are “well-connected”. Surprisingly, we find that the output of multiple clustering approaches—the Leiden algorithm with either the Constant Potts Model or modularity as quality function, Iterative K-Core Clustering, Infomap, and Markov Clustering—include communities that fail even a mild requirement for well-connectedness. As a remediation strategy, we have developed the “Connectivity Modifier” (CM), which iteratively removes small edge cuts and re-clusters until all communities detected are well-connected. Results from real-world networks with up to 75,025,194 nodes illustrate how CM enables additional insights into community structure within networks, while results on synthetic networks show that the CM algorithm improves accuracy in recovering true communities. Our study also raises questions about the “clusterability” of networks and mathematical models of community structure.

Keywords: community detection · connectivity · citation networks

1 Introduction

Community detection is of broad interest and is typically posed as a graph partitioning problem, where the input is a graph and the objective is a partitioning of its vertices into disjoint subsets, so that each subset represents a community [12, 23, 24]. The terms community and cluster overlap heavily, so we use them interchangeably herein. While community detection has many applications [8, 18], our interest is largely related to identifying research communities from the global scientific literature. Accordingly we are especially focused on methods that can scale to large citation networks [36, 37].

A general expectation is that the vertices within a community are better connected to each other than to vertices outside the community [7, 16], implying

M. Park, Y. Tabatabaee, and V. Ramavarapu—Contributed equally.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024

H. Cherifi et al. (Eds.): COMPLEX NETWORKS 2023, SCI 1142, pp. 3–14, 2024.

https://doi.org/10.1007/978-3-031-53499-7_1

greater edge density within a community. However, a cluster may be dense while still having a small min cut (the smallest edge set whose removal would disconnect the cluster) [3]. In other words, some dense clusters can be disconnected by small min cuts. Thus, *edge density* and *well-connectedness* are expected but *separable* properties of communities.

The potential for modularity optimization to produce poorly connected clusters, for example, two large cliques connected by a single edge, is well established [11, 36]. Less well studied, however, is the question of whether other clustering methods also produce poorly connected clusters. One of the commonly used clustering methods is the Leiden algorithm [36] optimizing the Constant Potts Model (CPM) [35]. Clusters produced by CPM-optimization with resolution parameter r have the desirable property that if an edge cut splits the cluster into components A and B , then the edge cut size will be at least $r \times |A| \times |B|$ (Supplementary Materials of reference [36]). This guarantee is strong when the edge cut splits a cluster into two components of approximately equal size, but is weaker when it produces an unbalanced split and weakest when the cut separates a single node from the remaining nodes in the cluster. Importantly, the guarantee depends on r , and small values of r produce weak bounds. It is also important to note that this guarantee applies to CPM-optimal clusterings but not to heuristics.

In using the Leiden software optimizing CPM, we observe that it produces clusters with small min cuts on seven different real-world networks of varied origin ranging in size from approximately 34,000 to 75 million nodes. We also observe that the number of clusters with small min cuts increases as the resolution parameter is decreased. Intrigued by this observation, we perform a broader study to evaluate the extent to which clusters produced by algorithms of interest meet even a mild standard for a well-connected cluster.

We formalize the notion of “poorly-connected” clusters by considering functions $f(n)$ with the interpretation that if a cluster of size n has an edge cut of size at most $f(n)$ then the cluster will be considered poorly connected. We want $f(n)$ to grow very slowly so that it serves as a mild bound. We also want $f(n) \geq 1$ for all n that are large enough for the cluster to be considered a potential community. We choose $f(n) = \log_{10} n$ from three examples of slow growing functions [25], since it imposes the mildest constraint on large clusters and grows more slowly than the bound for optimal CPM clusterings [36].

We examine min cut profiles from four additional clustering methods on the seven networks: Leiden optimizing modularity [24]; the k -core based Iterative k -core Clustering (IKC) [37]; and two flow-based methods, Infomap [31] and Markov Clustering (MCL) [9]. All the methods we tested produce poorly connected clusters on these networks, some produce tree clusters, and some even produce disconnected clusters. These observations reveal a gap between the expectation of well-connected clusters and what is actually being produced by these community finding methods.

To address this gap, we have developed the Connectivity Modifier (CM) [29] that takes a clustering as input and recursively removes small edge cuts and reclusters until all clusters are well-connected. Using CM on seven real-world

networks, we demonstrate the insights that CM can provide into community structure in networks. These findings also raise questions about the “clusterability” [22] of networks and whether only portions of a network exhibit community structure. Additional analyses on synthetic networks provide evidence that CM improves community detection accuracy under many conditions [25].

2 Results

2.1 Initial Observations

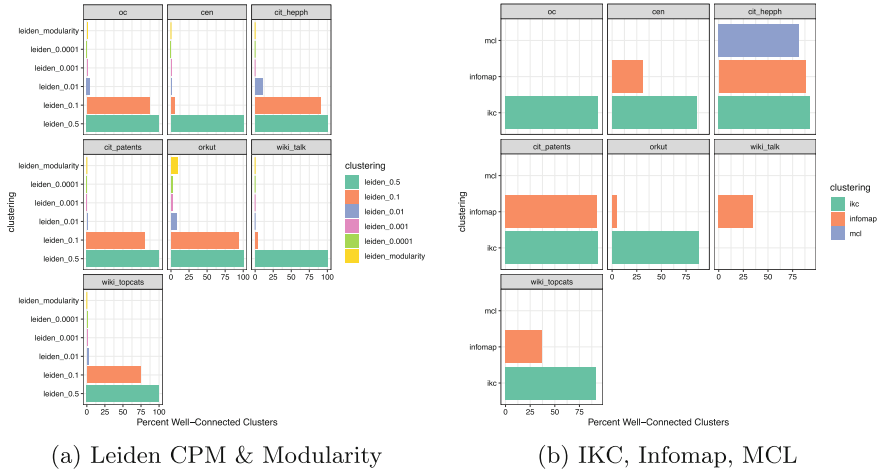


Fig. 1. *Percentage of well-connected clusters in seven real-world networks.* The networks studied range in size from 34,546 nodes to 75,025,194 nodes. Only Leiden and IKC ran to completion on all seven networks. Only Leiden-CPM with the largest tested resolution parameter (0.5) and IKC had 80% or more of their clusters considered well-connected. Five clustering methods were explored: (a) Leiden optimizing CPM at different resolution values and Leiden optimizing modularity, and (b) IKC, Infomap, and MCL. IKC did not return any clusters from the wiki_talk network. Infomap completed on all but Open Citations. MCL completed only on cit_hepph.

In an exploratory experiment, we clustered seven networks (Table 1, Materials and Methods), ranging in size from 34,546 nodes to 75,025,194 nodes, with Leiden, IKC, Infomap, and MCL, and computed the percentage of clusters whose min cuts were greater than $f(n)$. Under the conditions used, Leiden and IKC ran to completion on all seven networks, Infomap failed on the largest network, and MCL returned output only from the smallest network (cit_hepph) we analyzed.

This experiment (Fig. 1) revealed that all clustering methods generate clusters that are not well-connected, with the extent depending on the clustering

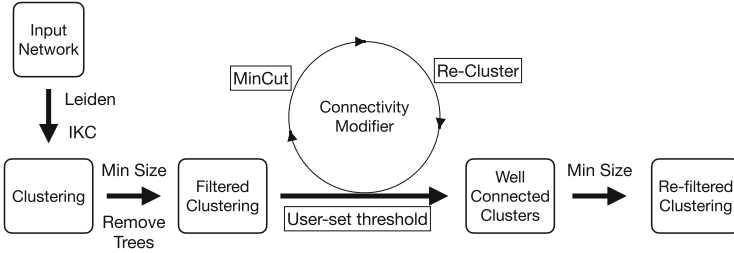


Fig. 2. *Connectivity Modifier Pipeline Schematic.* The four-stage pipeline depends on user-specified algorithmic parameters: B (default 11), the minimum allowed size of a cluster, and $f(n)$ (default $\log_{10}(n)$), a bound on the minimum edge cut size for a cluster with n nodes, and clustering method. *Stage 1:* a clustering is computed. *Stage 2:* clusters are pre-processed by removing trees and those clusters of size less than B . *Stage 3:* the CM is applied to each cluster, removing edge cuts of sizes at most $f(n)$, reclustering, and recursing on clusters. *Stage 4:* clusters are post-processed by removing those of size less than B .

method and network. Most significantly, only IKC and Leiden-CPM at a large resolution value returned a high fraction of well-connected clusters.

For Leiden clustering optimizing CPM, the frequency of well-connected clusters decreases with resolution value, and results from modularity are similar to the lowest resolution value for CPM that was tested. In comparison, nearly all IKC clusters were well-connected, with percentages that varied between 85.9% and 94% of the total number of clusters but with lower node coverage [25]. The percentage of well-connected clusters produced by Infomap varied from 5% (orkut) to 92.4% (cit_patents). For the single network that MCL completed on, 81.3% of the clusters were well-connected. Interestingly, both Infomap and MCL generated clusters that were disconnected, a limitation that had been previously noted for Louvain optimizing modularity [36].

2.2 Connectivity Modifier

To remediate poorly connected clusters, we developed the Connectivity Modifier (CM) [29], which takes a clustering as input and returns well-connected clusters. CM presently provides support for Leiden optimizing either CPM or modularity and IKC, the methods that scaled to the largest network we studied. CM is implemented in a pipeline (Fig. 2), which allows the user to specify two parameters: $f(n)$ (the bound on the size of a min cut) and B , the minimum allowed size of a cluster. In our study we explored $f(n) = \log_{10}(n)$ and $B = 11$, but the user can provide different settings. A pre-processing (filtering) step discards clusters that are trees or of size less than B , noting that any tree with ten or more nodes is not well-connected according to our definition of $f(n)$. CM then checks each cluster to see if it contains an edge cut of size at most $f(n)$, and if so CM removes the edge cut, following which the resultant subnetworks are reclustered. This process repeats until the current iteration produces no change.

A post-processing step removes any small clusters of size less than B that may have resulted from repeated cutting.

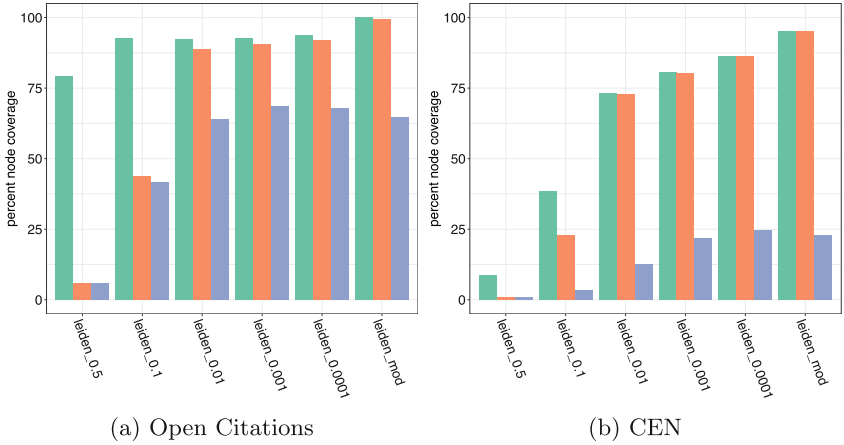


Fig. 3. Reduction in node coverage after CM treatment of Leiden clusters. The Open Citations (left panel) and CEN (right panel) networks were clustered using the Leiden algorithm under CPM at five different resolution values or modularity. Node coverage (defined as the percentage of nodes in clusters of size at least 2) was computed for (i) Leiden clusters (green), (ii) Leiden clusters with trees and clusters of size 10 or less filtered out (orange), and (iii) after CM treatment of filtered clusters (blue).

To further understand the nature of the modifications effected by CM, we also classified the Leiden clusters based on the impact of CM-processing: *extant*, *reduced*, *split*, and *degraded*, where “extant” indicates that the cluster was not modified by CM, “reduced” indicates that the cluster is reduced in size, “split” indicates that the cluster was divided into at least two smaller clusters, and “degraded” indicates that the cluster was reduced to singletons or a cluster of size 10 or less [25]. All methods produced split clusters, suggesting “resolution limit” behavior that has already been documented for modularity [11]. Our study shows this also occurs at some non-negligible frequency for CPM-optimization using small resolution values, as well as for the other clustering methods.

2.3 Effect of CM on Clustered Real World Networks

We studied the effect of CM on clusterings generated by Leiden-modularity, Leiden-CPM, and IKC, the only methods that scaled to the largest network we studied and did not produce disconnected clusters. We present results here from the Open Citations and CEN networks, the two largest networks out of seven studied. Results on the remaining five networks show similar trends [25]. In assessing the impact of CM on node coverage, we do not consider very small clusters ($n \leq 10$) of practical interest, therefore, unless otherwise described, node

coverage is reported in this submission as the percentage of nodes in clusters of at least size 11.

With Leiden, maximum node coverage was 24.6% for the CEN and 68.7% for Open Citations for six conditions tested. Post-CM node coverage was smallest when using CPM-optimization with resolution value $r = 0.5$ and largest when optimizing CPM with one of the two smallest resolution values, $r = 0.001$ for CEN and $r = 0.0001$ for Open Citations. In comparison to Leiden, the IKC-clustering results in relatively low node coverage, 23.6% and 3.8% in the case of the Open Citations and CEN networks respectively. CM treatment of these clusterings has a small effect on node coverage. Results for the other five networks show similar trends and are available at [25].

The impact of CM depends on both the clustering method and the input network. A more detailed examination shows the following trends. For CPM-optimization, the impact of pre-processing is large for the two larger resolution values, but decreases with resolution value. In contrast, the step of the CM pipeline that iteratively finds and removes small edge cuts and reclusters has a minimal impact at larger resolution values and an increasing impact as the resolution value decreases. Modularity returned results most similar to CPM-optimization with the smallest tested resolution value.

These trends suggest the possibility that only a fraction of the nodes in a network belong in clusters that are sufficiently well-connected and sufficiently large. In other words, and through the lens of our mild standard for connectivity and size, real-world networks may not be fully covered by “valid” communities, in contrast to what is often assumed in mathematical models of community structure in networks.

2.4 Synthetic LFR Networks

To evaluate the impact of CM-processing on accuracy, we examined synthetic networks produced using LFR software [19]. For this experiment, we computed statistics for the Leiden clusterings of the seven real-world networks we previously explored, and used them as input to the LFR software [19] (see Materials and Methods). We produced a collection of 34 LFR networks with ground truth communities and clustered each of these 34 LFR networks using Leiden with the same clustering parameters used to provide empirical statistics to LFR.

We examined the impact of CM-processing on clustering accuracy; results for Normalized Mutual Information (NMI) and Adjusted Mutual Information (AMI) are shown in Fig. 4. Adjusted Rand Index (ARI) data are available in supplementary data [25]. CM-processing improves NMI accuracy for modularity and also for CPM-optimization when used with small resolution values. CM-processing tends to be otherwise neutral for NMI, and was only detrimental in two cases. The impact on AMI accuracy is more variable. For example, CM-processing reduced AMI accuracy for all conditions with `wiki_talk` except for CPM-optimization with $r = 0.1$, where accuracy was very low and the impact was neutral. CM-processing also reduced AMI accuracy for CPM-optimization on some network:clustering pairs for large resolution values. However, the cases

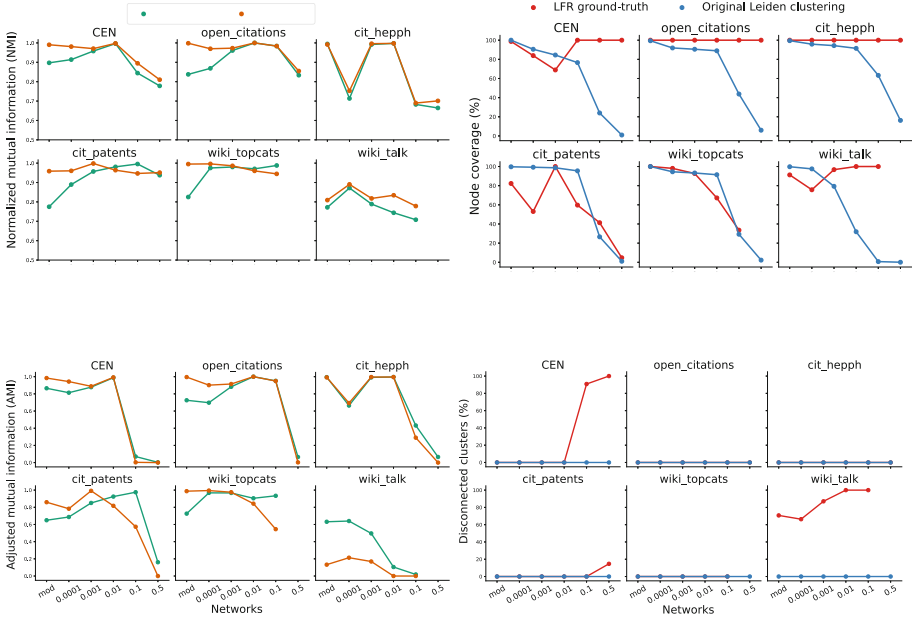


Fig. 4. *Impact of CM-processing on accuracy of synthetic networks.* The left panels show accuracy measured in terms of NMI and AMI with respect to the LFR ground-truth communities. Each condition on the x-axis corresponds to a *different* LFR network, generated based on Leiden-modularity or Leiden-CPM with that specific resolution parameter. The right panels show (top) node coverage when restricted to clusters of size at least 11, and (bottom) percent of clusters that are disconnected. Under most conditions, CM improves the accuracy of the original Leiden clustering, except when the ground-truth communities have many (at least 60%) disconnected clusters, or the node coverage by clusters of size at least 11 is relatively low (at most 70%).

where CM-processing produced a noteworthy reduction in accuracy for NMI or AMI are those where there are many disconnected ground truth clusters or where there are many clusters of size less than 11 in the “ground-truth clusters”. Interestingly, CM can still have a positive impact even when there are many small clusters or some disconnected clusters.

It is easy to see why low node coverage by clusters of size at least 11 could reduce accuracy for CM-processing, since CM with $B=11$ automatically removes all clusters below size 11. Therefore, CM-processing will not be beneficial where there is interest in recovering small communities unless the bound B is replaced by a smaller value. In contrast, the occurrence of disconnected ground-truth clusters in the LFR networks is striking and problematic, since an expectation of a community is that it is connected [36]. Hence, we assert that it is unreasonable to evaluate accuracy with respect to a ground-truth set of communities if the communities are not connected. The fact that LFR networks had ground truth clusters that were not connected also indicates the failure of LFR software to

reproduce features of the input network:clustering pairs, which by construction always have 100% of the clusters connected.

Once the comparison between CM-processed clusters and original clusters is restricted to the LFR networks, which do not have disconnected clusters and which have high node coverage after restriction to clusters of size at least 11), we find that CM ranges from neutral to beneficial in terms of the impact on accuracy. Thus, there is a potential benefit in using CM to modify clusters that are not well-connected.

3 Discussion

In this study we report that multiple clustering methods produce clusters that do not meet a mild standard for well-connectedness. To enforce user-defined standards for connectivity and cluster size, we developed the Connectivity Modifier (CM), an open-source tool [29] that presently provides support for the Leiden and IKC software. We show that the CM enables users to assess hidden community structure within clusters.

How significantly CM-processing changes a given clustering depends on the network itself, as some networks seem to be more impacted by CM-processing. It also depends on the choice of clustering algorithm and the parameter settings used.

An important implication of our study using CM on seven real world communities is the possibility that portions of real world networks may not exhibit robust community structure. A related study using a different approach [22] considers the question of whether a graph can be considered clusterable. Moreover, since many clustering methods aim to maximize node coverage, such approaches are likely to merge weakly connected parts of a network into communities that fail modest standards for connectivity. Thus, there is a trade-off between node coverage and well-connectedness that the user must consider in exploring community structure.

The finding that LFR networks produce different patterns than empirical networks is not surprising. First, the LFR methodology assumes that the degree distribution and cluster size distributions follow a power law, which may not apply well to many real-world networks [2, 5, 28, 33]. Second, the degree distribution and cluster size distributions were imperfectly fitted by LFR software in our study. Another questionable assumption in the LFR methodology is that every node is in a community; our study suggests that this assumption may only be reasonable if the communities can be small and/or poorly connected. Exploring the value of improved generators such as *ABCD_o* [17] is a next step.

Several questions remain unanswered. First, to be able to distinguish between the existence of a community and the ability to detect it with a given clustering method. Second, to assess whether too many or too few nodes are being dropped. While we emphasize leaving the definition, use, and interpretation of well-connected to users, these questions merit attention. We also recognize that our emphasis on well-connected clusters may result in narrow descriptions of

communities. Additionally, informative weaker links [13] may be lost from communities since CM partitions input clusters that are poorly connected into sets of well-connected clusters. On the other hand, these well-connected communities could be useful in defining cores of core-periphery structures [4, 30, 37] that could subsequently be augmented by adding peripheral components. Last, future work should incorporate evaluation criteria relevant to the input network.

Materials and Methods

Table 1. Real world networks used in this study

| network | # nodes | # edges | average_node_deg | reference |
|----------------|------------|---------------|------------------|-----------|
| Open Citations | 75,025,194 | 1,363,303,678 | 36.34 | [27] |
| CEN | 13,989,436 | 92,051,051 | 13.16 | [15] |
| cit_hepph | 34,546 | 420,877 | 24.37 | [21] |
| cit_patents | 3,774,768 | 16,518,947 | 8.75 | [21] |
| orkut | 3,072,441 | 117,185,083 | 76.28 | [38] |
| wiki_talk | 2,394,385 | 4,659,565 | 3.89 | [20] |
| wiki_topcats | 1,791,489 | 25,444,207 | 28.41 | [39] |

Data. The publicly available Open Citations dataset was downloaded in Aug 2022. The CEN is a citation network constructed from the literature on exosome research. From the SNAP repository, we downloaded cit_hepph, an High Energy Physics citation network; cit_patents, a citation network of US patents; orkut, a social media network; wiki_talk, a network containing users and discussion from the inception of Wikipedia until January 2008; and wiki_topcats, a web graph of Wikimedia hyperlinks. All networks were processed to remove self-loops, duplicate edges, and parallel edges before clustering.

LFR (Synthetic) Networks. To create simulated networks with ground truth communities, we used the LFR software [10, 19], which takes the following eight parameters as input:

- Network properties: Number of nodes N , average and maximum node degrees (k and k_{max} respectively), and negative exponent for degree sequence (τ_1).
- Community properties: Maximum and minimum community sizes (c_{max} and c_{min}), and negative exponent for the community size distribution (τ_2).
- Mixing parameter μ , that is the ratio between the degree of a node outside its community and its total degree, averaged over all nodes in the network.

To emulate the empirical networks using LFR graphs, we estimated all eight parameters described above for a given pair of network \mathcal{G} and a clustering \mathcal{C} . N, k, k_{max}, c_{min} and c_{max} were computed using *networkX* [14, 34]. To estimate

μ , we performed a single iteration over all edges of the network, and for each edge, if the nodes on the two sides of it were in different communities, that edge contributes to the ratio μ of these two nodes. The total μ of the network:clustering pair is the average μ across all the nodes.

To estimate τ_1 and τ_2 , we fitted a power-law distribution to the node degree sequence and the community size distribution, using the approach from [6] that is implemented in the *powerlaw* Python package [1]. Because the power-law property may not hold for the whole distribution, following [6], we estimated x_{min} , the minimum value for which the power-law property holds as well as the exponent α for the tail of the distribution.

After computing these parameters based on the Leiden clusterings of the empirical networks using both modularity and CPM with a range of resolution parameters, we simulated LFR networks [19]. For networks with more than 10 million nodes, i.e., Open Citations and the CEN, we limited the number of vertices to 3 million, due to scalability limitations of the LFR benchmark graph generator [32], while preserving the edge density reflected by average degree, and the mixing parameter. The numbers of nodes of the other LFR graphs exactly match the number of nodes in the corresponding empirical network. In some cases, due to the inherent limitations of the LFR graph generator, we had to modify the ranges of the community sizes, i.e., increase c_{min} and decrease c_{max} , to generate the network. We calculated NMI, AMI, and ARI using the Python Scikit-Learn package [26].

Acknowledgments and Funding. The authors acknowledge funding from the InSpier-Illinois partnership and an Oracle Research Award to TW. We thank three anonymous reviewers for very constructive critique.

References

1. Alstott, J., Bullmore, E., Plenz, D.: *powerlaw*: a Python package for analysis of heavy-tailed distributions. *PloS ONE* **9**(1), e85,777 (2014)
2. Artico, I., Smolyarenko, I., Vinciotti, V., Wit, E.C.: How rare are power-law networks really? *Proc. Roy. Soc. A* **476**(2241), 20190,742 (2020)
3. Bonchi, F., García-Soriano, D., Miyauchi, A., Tsourakakis, C.E.: Finding densest k -connected subgraphs. *Discret. Appl. Math.* **305**, 34–47 (2021)
4. Breiger, R.: *Explorations in Structural Analysis (RLE Social Theory)*. Routledge, Milton Park (2014)
5. Brzezinski, M.: Power laws in citation distributions: evidence from Scopus. *Scientometrics* **103**, 213–228 (2015)
6. Clauset, A., Shalizi, C.R., Newman, M.E.: Power-law distributions in empirical data. *SIAM Rev.* **51**(4), 661–703 (2009)
7. Coscia, M., Giannotti, F., Pedreschi, D.: A classification for community discovery methods in complex networks. *Stat. Anal. Data Min.* **4**(5), 512–546 (2011)
8. Dey, A.K., Tian, Y., Gel, Y.R.: Community detection in complex networks: from statistical foundations to data science applications. *WIREs Comput. Stat.* **14**(2) (2021). <https://doi.org/10.1002/wics.1566>

9. Dongen, S.V.: Graph clustering via a discrete uncoupling process. *SIAM J. Matrix Anal. Appl.* **30**(1), 121–141 (2008)
10. Fortunato, S.: Resources (2023). <https://www.santofortunato.net/resources>
11. Fortunato, S., Barthelemy, M.: Resolution limit in community detection. *Proc. Natl. Acad. Sci.* **104**(1), 36–41 (2007)
12. Fortunato, S., Newman, M.E.J.: 20 years of network community detection. *Nat. Phys.* **18**(8), 848–850 (2022)
13. Granovetter, M.S.: The strength of weak ties. *Am. J. Sociol.* **78**(6), 1360–1380 (1973)
14. Hagberg, A., Swart, P., S Chult, D.: Exploring network structure, dynamics, and function using NetworkX. Technical report, Los Alamos National Laboratory (2008)
15. Jakatdar, A., Liu, B., Warnow, T., Chacko, G.: AOC: assembling overlapping communities. *Quant. Sci. Stud.* **3**(4), 1079–1096 (2022)
16. Javed, M.A., Younis, M.S., Latif, S., Qadir, J., Baig, A.: Community detection in networks: a multidisciplinary review. *J. Netw. Comput. Appl.* **108**, 87–111 (2018)
17. Kamiński, B., Pralat, P., Thériberge, F.: Outliers in the ABCD random graph model with community structure (ABCDo). In: Cherifi, H., Mantegna, R.N., Rocha, L.M., Cherifi, C., Micciche, S. (eds.) *COMPLEX NETWORKS 2016 2022*. *SCI*, vol. 1078, pp. 163–174. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-21131-7_13
18. Karataş, A., Şahin, S.: Application areas of community detection: a review. In: *2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT)*, pp. 65–70. IEEE (2018)
19. Lancichinetti, A., Fortunato, S., Radicchi, F.: Benchmark graphs for testing community detection algorithms. *Phys. Rev. E* **78**(4), 046,110 (2008)
20. Leskovec, J., Huttenlocher, D., Kleinberg, J.: Signed networks in social media. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1361–1370. ACM (2010)
21. Leskovec, J., Kleinberg, J., Faloutsos, C.: 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*, pp. 177–187. ACM (2005). <https://doi.org/10.1145/1081870.1081893>
22. Miasnikof, P., Shestopaloff, A.Y., Raigorodskii, A.: Statistical power, accuracy, reproducibility and robustness of a graph clusterability test. *Int. J. Data Sci. Anal.* **15**(4), 379–390 (2023)
23. Mucha, P.J., Richardson, T., Macon, K., Porter, M.A., Onnela, J.P.: Community structure in time-dependent, multiscale, and multiplex networks. *Science* **328**(5980), 876–878 (2010)
24. Newman, M.E., Girvan, M.: Finding and evaluating community structure in networks. *Phys. Rev. E* **69**(2), 026,113 (2004)
25. Park, M., et al.: Well-connected communities in real-world and synthetic networks (2023). <https://tandy.cs.illinois.edu/cm-for-webpage-combined.pdf>. Includes supplementary materials
26. Pedregosa, F., et al.: Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011)
27. Peroni, S., Shotton, D.: OpenCitations, an infrastructure organization for open scholarship. *Quant. Sci. Stud.* **1**(1), 428–444 (2020)
28. Radicchi, F., Fortunato, S., Castellano, C.: Universality of citation distributions: toward an objective measure of scientific impact. *Proc. Natl. Acad. Sci.* **105**(45), 17268–17272 (2008)

29. Ramavarapu, V., Ayres, F., Park, M., Pailodi, V.K., Chacko, G., Warnow, T.: Connectivity modifier (2023). https://github.com/illinois-or-research-analytics/cm_pipeline
30. Rombach, P., Porter, M.A., Fowler, J.H., Mucha, P.J.: Core-periphery structure in networks (revisited). *SIAM Rev.* **59**(3), 619–646 (2017)
31. Rosvall, M., Bergstrom, C.T.: Maps of random walks on complex networks reveal community structure. *Proc. Natl. Acad. Sci.* **105**(4), 1118–1123 (2008)
32. Slota, G.M., Berry, J.W., Hammond, S.D., Olivier, S.L., Phillips, C.A., Rajamanickam, S.: Scalable generation of graphs for benchmarking HPC community-detection algorithms. In: *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–14 (2019)
33. Stringer, M.J., Sales-Pardo, M., Amaral, L.A.N.: Statistical validation of a global model for the distribution of the ultimate number of citations accrued by papers published in a scientific journal. *J. Am. Soc. Inform. Sci. Technol.* **61**(7), 1377–1385 (2010)
34. Tabatabaee, Y.: Emulating real networks using LFR graphs (2023). <https://github.com/ytabatabaee/emulate-real-nets>
35. Traag, V.A., Dooren, P.V., Nesterov, Y.: Narrow scope for resolution-limit-free community detection. *Phys. Rev. E* **84**(1) (2011)
36. Traag, V.A., Waltman, L., Van Eck, N.J.: From Louvain to Leiden: guaranteeing well-connected communities. *Sci. Rep.* **9**(1), 1–12 (2019)
37. Wedell, E., Park, M., Korobskiy, D., Warnow, T., Chacko, G.: Center-periphery structure in research communities. *Quant. Sci. Stud.* **3**(1), 289–314 (2022)
38. Yang, J., Leskovec, J.: Defining and evaluating network communities based on ground-truth. *Knowl. Inf. Syst.* **42**(1), 181–213 (2013)
39. Yin, H., Benson, A.R., Leskovec, J., Gleich, D.F.: Local higher-order graph clustering. In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 555–564. ACM (2017)