From Local to Global Communities in Large Networks Through Consensus*

Mariano Tepper and Guillermo Sapiro

ECE, Duke University, USA. {mariano.tepper, guillermo.sapiro}@duke.edu

Abstract. Given a universe of local communities of a large network, we aim at identifying the meaningful and consistent communities in it. We address this from a new perspective as the process of obtaining *consensual* community detections and formalize it as a bi-clustering problem. We obtain the *global* community structure of the given network without running expensive global community detection algorithms. The proposed mathematical characterization of the consensus problem and a new bi-clustering algorithm to solve it render the problem tractable for large networks. The approach is successfully validated in experiments with synthetic and large real-world networks, outperforming other state-of-the-art alternatives in terms of speed and results quality.

1 Introduction

The inference of global community structure in networks (i.e., finding groups of nodes such that intra-group connections are denser than inter-group ones) has become a topic of great interest in the data analysis scientific community [4,5,15,24]. However, the best way to establish the community structure is still disputed [7]. This is particularly true for large networks, as the vast majority of community detection algorithms, e.g., modularity maximization [15], require the network to be completely known. A family of algorithms aims at finding communities by analyzing the network locally [1,4,22,24]. But how can we go from a collection of such *local* communities to *global* community structure? Addressing this efficiently for large networks is the topic of this work.

Let $G = (V, E, \psi)$ be the graph to analyze, where V is the set of m nodes, E is the set of edges, and $\psi : E \to \mathbb{R}^+$ is a weighting function on the edges (in the following we use the terms graph and network interchangeably). Generically, we consider that a local community-detection algorithm provides a candidate community $C \subset V$. Let us consider that we are provided with a pool (universe) $\{C_k\}_{k=1}^n$ of n such candidates. These candidates might come from a combination of running **(a)** different local community-detection algorithms, **(b)** one local algorithm with different parameters or initializations, and/or **(c)** one local algorithm on different modalities of the same network, by changing the set of edges and/or the function ψ (this case includes a network that changes over time).

* Work partially supported by NSF, ONR, NGA, ARO, and NSSEFF.

Given the pool $\{C_k\}_{k=1}^n$, how do we stitch these local communities together to obtain a *coherent* and *global* structure of the network G? Consensus/ensemble clustering is a well known family of techniques used in data analysis to solve this type of problem when we have a pool of partitions (instead of single communities as we have here). Typically, the goal is to search for the partition that is most similar, on average, to all input partitions. See [21] for a survey of the subject.

The most common form of consensus clustering for the case of a pool of single communities involves creating an $m \times m$ co-occurrence matrix $\mathbf{B}$, such that

$$\mathbf{B} = \tfrac{1}{c} \sum_{i=1}^{n} \mathbf{B}_k \qquad \text{where} \qquad (\mathbf{B}_k)_{ij} = \begin{cases} 1 & \text{if } i,j \in C_k; \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

There are many algorithms for analyzing $\mathbf{B}$, from simple techniques such as applying a clustering algorithm to it (e.g., k-means or hierarchical clustering), to more complex techniques [13]. Consensus for community detection was addressed in [3] and [8] within the standard formulation just described.

The mentioned aggregation process used to build $\mathbf{B}$ involves losing information contained in the individual matrices $\mathbf{B}_k$. In particular, only pairwise relations are conserved, while relations involving larger groups of nodes might be lost. In addition, using the average of several partitions might not be robust if some of them are of poor quality.

Contributions. We propose a novel formal framework and perspective for reaching consensus community detection by posing it as a bi-clustering problem [20]. We also introduce a new bi-clustering algorithm, fit for the type of matrices we analyze. The core novelty of this work is the use of a consensus framework for stitching local communities together, produced by algorithms that do not necessarily analyze large/huge networks as a whole. This allows to integrate this partial and seemingly disaggregated information into a coherent and global structural description of large networks. Our framework does not require a community quality measure to make decisions but can use of it if it is available.

Organization. In Section 2 we present the proposed approach. In Section 3 we discuss the experimental results. We provide some closing remarks in Section 4.

2 Consensus Community Detection

The input of the consensus algorithm is a pool $\mathcal{U} = \{C_k\}_{k=1}^n$ of candidates. We also assign a weight $w_k \in \mathbb{R}^+$ to each community candidate C_k. From V (the set of nodes) and $\mathcal{U}$, we define the $m \times n$ preference matrix $\mathbf{A}$. The element $(\mathbf{A})_{ik} = w_k$ if the i-th node belongs to the k-th local community, and 0 otherwise.

The community weights indicate the importance assigned to each candidate and can take any form. The simplest form uses uniform weights $(\forall k)\ w_k = 1$, in which case $\mathbf{A}$ becomes a binary matrix. In this case, no prior information is used about the quality of the input community candidates. If we have such information, it can be freely incorporated in these weights.

We are interested in finding clusters in the product set $V \times \mathcal{U}$. The main contribution of this work is to establish the formal connection between bi-clustering

the preference matrix $\mathbf{A}$ (as a complete representation of $\mathcal{U}$) and the extraction of global community structure from local communities. This provides a very intuitive rationale: for each bi-cluster, we are jointly selecting a subset of nodes and local communities such that the former belongs to the latter.

2.1 Solving the bi-clustering problem

Among many tools for bi-clustering, see [16] and references therein, the Penalized Matrix Decomposition [23] and the Sparse Singular Value Decomposition [12] have shown great promise, mainly due to their conceptual and algorithmic simplicity. Correctly setting the parameters of these methods is crucial, since they determine the size of the bi-clusters [12,18,23]. In our experiments, finding the correct values for these parameters has proven extremely challenging, since each experiment needs a specifically tuned set of values.

We propose to follow a different path for solving the bi-clustering problem at hand. For $1 \leq q \leq \min\{m, n\}$, we define

$$\min_{\mathbf{X}\in\mathbb{R}^{m\times q},\mathbf{Y}\in\mathbb{R}^{q\times n}} \|\mathbf{A} - \mathbf{X}\mathbf{Y}\|_1 \quad \text{s.t.} \quad \mathbf{X}, \mathbf{Y} \geq 0. \tag{L1-NMF}$$

$\mathbf{A}$ is, in our application, a sparse non-negative matrix. Notice that the positivity constraints on $\mathbf{X}, \mathbf{Y}$ have a sparsifying effect on them. The intuition behind this is that when approximating a sparse non-negative matrix, the non-negative factors will only create a sparse approximation if they are themselves sparse. We thus obtain sparse factors $\mathbf{X}, \mathbf{Y}$ without introducing any (difficult to set) parameters. With the L1 fitting term, we are aiming at obtaining a "median" type of result instead of the mean, providing robustness to poor group candidates present in the pool (i.e., spurious columns of $\mathbf{A}$).

A challenge with NMF is that q is not an easy parameter to set. To avoid a cumbersome decision process, we propose to set $q = 1$ and inscribe the L1-NMF approach in an iterative loop, find one rank-one at a time. The rank-one factorization $\mathbf{X}\mathbf{Y}$ will thus approximate a subset of $\mathbf{A}$ (because of the sparsity-inducing L1-norm), correctly detecting a single bi-cluster. Let $\mathcal{R}, \mathcal{Q}$ be the active sets (non-zero entries) of $\mathbf{X}$, $\mathbf{Y}$, respectively: $\mathcal{R}$ selects rows (nodes), while $\mathcal{Q}$ selects columns (local communities).

Algorithm 1 summarizes the proposed non-negative bi-clustering approach. Notice that instead of subtracting the product $\mathbf{X}\mathbf{Y}$ from $\mathbf{A}$, we set the corresponding rows and columns to zero, enforcing disjoint active sets between the successive $\mathbf{X}_t$ and $\mathbf{Y}_t$, and hence orthogonality. This also ensures that non-negativity is maintained throughout the iterations. If the bi-clusters are allowed to share nodes, we do not change the rows of $\mathbf{A}$. This is an important feature for community detection since overlapping communities are ubiquituous. The proposed algorithm is very efficient, simple to code, and demonstrated to work well in the experimental results that we will present later.

The iterations should stop (1) when $\mathbf{A}$ is empty (line 2), or (2) when $\mathbf{A}$ contains no structured patterns. The second case is controlled by τ_{R} and τ_{C} (line 6), which determine the minimum size that a bi-cluster should have. Note

Algorithm 1: Bi-clustering algorithm.

input : Preference matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, stopping parameters τ_{R} and τ_{C}.
output: Bi-clusters $\{(\mathcal{R}_t, \mathcal{Q}_t)\}_{t=1}^{T}$

$t \leftarrow 0$;
while $\mathbf{A} \neq \mathbf{0}$ **do**
  $t \leftarrow t + 1$;
  Solve Problem (L1-NMF) for $\mathbf{X}_t, \mathbf{Y}_t$ with $q = 1$;
  $\mathcal{R}_t \leftarrow \{i \mid i \in [1, m],\ (\mathbf{X}_t)_{i,1} \neq 0\}$; $\mathcal{Q}_t \leftarrow \{j \mid j \in [1, n],\ (\mathbf{Y}_t)_{1,j} \neq 0\}$;
  if $|\mathcal{R}_t| \leq \tau_R \vee |\mathcal{Q}_t| \leq \tau_C$ **then** $t \leftarrow t - 1$; **break**;
  if *non-overlapping biclusters are desired* **then**
    $(\forall i,j)\ i \in \mathcal{R}_t, j \in [1, n],\ (\mathbf{A})_{ij} \leftarrow 0$;
  $(\forall i,j)\ i \in [1, m], j \in \mathcal{Q}_t,\ (\mathbf{A})_{ij} \leftarrow 0$;
$T \leftarrow t$;

that these parameters are intuitive, related to the physics of the problem, and easy to set. In all experiments in this paper, we set $\tau_{\mathrm{R}} = 3$ and $\tau_{\mathrm{C}} = 2$.

Implementation and scalability. Any NMF algorithm can be adapted to use the L1 norm and solve (L1-NMF); in this work, we use the method in [20]. Following [14], it is relatively straightforward to implement any NMF algorithm in a MapReduce framework, making it completely scalable.

2.2 Extracting local communities

For community detection in large networks, algorithms that analyze the network locally are becoming popular [1,4,22,24]. These algorithms are extremely fast and can be adapted to optimize a wide range of different local quality measures. However this partial and seemingly disaggregated information does not provide a description of the full large network. We use our framework to integrate these local communities, obtaining such a coherent and global structural description.

In particular, given a single member s of an unknown community S, we aim at discovering S itself. To address this task, we use the PageRank-Nibble method [1,24], mainly because of its efficiency. Its computational cost is independent of the network size and proportional to the size of the detected community (think about how many friends you have in Facebook versus the total number of Facebook users). The key idea of PageRank-Nibble is to obtain a local spectral clustering algorithm by computing random walks starting from a single seed node s [1]. The PageRank scores give a measure of how well connected are the nodes around s. Algorithm 2 summarizes the overall procedure.

Conductance is the scoring function of choice in [24]. Communities are identified as local minima of $\{f(S_1) \dots f(S_h)\}$. In our experiments, we found that conductance does not produce robust local minima; we use an alternative function. Modularity [15], the standard community scoring measure, is not effective for assessing small communities in large networks [7]. However, if we restrict the computation to the subgraph induced by the set $\{u \in V \mid r_u \geq \varepsilon\}$, modularity

Algorithm 2: Community detection from a seed node

input : Graph $G(V,E)$, seed node s, community scoring function f, threshold ε.
output: A community $S_{k_{\text{ext}}} \subset V$ containing the seed node s.

1 Compute random walk scores r_u from seed node s using PageRank-Nibble [1];
2 Compute the sweep vectors set $\{S_k\}_{k=1}^{h}$ [1], where $S_k = S_{k-1} \cup \{\text{argmax}_{u \notin S_{k-1}, r_u \geq \varepsilon} \frac{r_u}{d(u)}\}$, $S_1 = s$, and $d(u)$ is the degree of u;
3 Find the index k_{ext} corresponding to the extremum of $\{f(S_1) \dots f(S_h)\}$;

becomes effective. We name this measure local modularity. This choice is not critical in our framework and any other suitable scoring function can be used instead, e.g. [4].

In our experiments, we extract a single community per seed (notice again that this is not critical in our framework). We thus identify the local community around a seed node s as the sweep vector with maximum local modularity.

The pool of candidates is the set of all local communities, using every node as a seed. Alternatively, we could randomly sample the set of nodes, in a Monte Carlo fashion. Moreover, any local community detection algorithm, e.g., [4,22], or even a combination of several of them, can be used with our framework. In this work, we use a binary preference matrix, i.e., uniform weights.

3 Experimental Results

We begin by testing the proposed framework in synthetic networks with ground truth communities, see Table 1. The performance of the consensus solution is on par with what is achieved by analyzing the ground truth and then picking the best local communities in the pool of candidates. The consensus community structure is obtained without tuning parameters nor accessing the ground truth.

We tested the proposed approach on four real-world networks (Amazon, DBLP, Youtube, and Orkut) with a functional definition of ground-truth [24] (`http://snap.stanford.edu/`). The ground-truth contains the top 5,000 ground-truth communities of each network. To run the simulations with these large networks, we accelerate our computations by pruning, from the preference matrix, the columns (local communities) whose local modularity is below a threshold γ (for Amazon, DBLP, Youtube, and Orkut $\gamma = 0.22, 0.2, 0.22, 0.09$, respectively).

We compare the proposed approach with other state-of-the-art methods both in terms of speed and quality of the results. The stochastic variational inference (SVI) algorithm [6] was particularly designed to work with large networks. However, the number of communities is an input to the method and the author's code does not scale well with this quantity: trying to discover more than 500 communities led to out-of-memory issues. We thus set this number to 500 for all experiments. The global consensus algorithm (LF) in [8] allows to use different base community detection algorithms; we used OSLOM [11], Infomap [19], Louvain [2], and Label Propagation (LP) [17]. All tests were performed on a desktop computer with an Intel i7-4770 CPU, 32 GB of RAM, and Ubuntu Linux 12.04.

Table 1: Results with synthetic networks (m is the number of nodes and δ the average node degree), produced with a standard benchmark generator [10]. In each experiment, we compute generalized normalized mutual information scores [9] over 100 different synthetic instances. For this comparison, we created a small subset of the local communities that fit more closely each of the T_{GT} ground truth communities. By picking the best and the median in this subset we obtain each of the T_{GT} best local communities and each of the T_{GT} good local communities, respectively. The consensus solution is always competitive with the best base solution and outperforms the good base solution, both found by using the ground truth and thus providing idealized performances (hardly achievable in practice). Our result is achieved by only looking at the base local communities.

		Mean	STD	Median	Min	Max
$m = 10^2, \delta = 10$	Consensus	0.879	0.171	0.964	0.336	1.000
	T_{GT} best local communities	0.872	0.126	0.926	0.446	1.000
	T_{GT} good local communities	0.654	0.159	0.676	0.256	0.924
$m = 10^2, \delta = 15$	Consensus	0.705	0.217	0.707	0.289	1.000
	T_{GT} best local communities	0.733	0.198	0.779	0.239	1.000
	T_{GT} good local communities	0.664	0.175	0.701	0.255	0.917
$m = 10^3, \delta = 10$	Consensus	0.812	0.053	0.818	0.665	0.922
	T_{GT} best local communities	0.844	0.036	0.850	0.749	0.924
	T_{GT} good local communities	0.790	0.047	0.795	0.648	0.879
$m = 10^3, \delta = 15$	Consensus	0.661	0.066	0.659	0.536	0.841
	T_{GT} best local communities	0.683	0.041	0.681	0.566	0.798
	T_{GT} good local communities	0.611	0.044	0.609	0.506	0.713
$m = 10^4, \delta = 30$	Consensus	0.618	0.067	0.622	0.332	0.773
	T_{GT} best local communities	0.455	0.070	0.457	0.171	0.633
	T_{GT} good local communities	0.307	0.075	0.306	0.117	0.577

Our method outperforms the alternatives in all the evaluated networks, see Table 2. In the Amazon network, our method is 3 minutes slower than LF+Louvain but obtains a performance gain of 34% over it; it is one order of magnitude faster than LF+Infomap with a performance gain of 8%. These comparisons are much more favorable to our method in the DBLP network. Our method is 4 times faster and almost 3 times better than LF+Louvain; it is also 68 times faster than LF+OSLOM with a performance boost of 48%.

Figure 1 presents two examples of the results of our bi-clustering algorithm. First, we observe that the proposed approach clearly organizes the data contained in the preference matrix, detecting an almost block-diagonal structure in it (albeit with overlaps and a small amount of noise in the preference matrix itself). Another important observation is that the detected bi-clusters share nodes (see the zoomed-in details), thus detecting overlapping consensus communities.

4 Conclusions

We analyzed for the first time the process of generating a global community structure from locally-extracted communities and formalized it as a bi-clustering consensus problem. This offers a new perspective for this important problem. Our consensus algorithm stitches the local communities together, obtaining a

Table 2: Comparison of the efficiency and accuracy of the proposed approach against other community detection methods. We compare with the stochastic inference algorithm (SVI) [6], and with the *global* consensus algorithm (LF) [8], using different base community detection algorithms. Our approach achieves top precision performances in minimum time (respectively measured as the fraction of retrieved communities that the ground truth and in minutes). V and E are the set of vertices and edges of the networks, respectively.

Algorithms	Amazon $\#V \approx 3 \times 10^6$ $\#E \approx 9 \times 10^6$		DBLP $\#V \approx 3 \times 10^6$ $\#E \approx 1 \times 10^7$		Youtube $\#V \approx 1 \times 10^7$ $\#E \approx 3 \times 10^7$		Orkut $\#V \approx 3 \times 10^7$ $\#E \approx 1 \times 10^9$	
	Time	Prec.	Time	Prec.	Time	Prec.	Time	Prec.
LF [8] + OSLOM [11][1]	263.15	0.837	660.68	0.664	–[3]	–	–[3]	–
LF [8] + Infomap [19][1]	179.83	0.900	407.43	0.616	–[3]	–	–[3]	–
LF [8] + Louvain [2][1]	8.10	0.724	41.55	0.347	–[4]	–	–[3]	–
LF [8] + LP [17][1]	23.37	0.792	172.40	0.406	–[3]	–	–[3]	–
SVI [6][2]	58.88	0.409	28.22	0.388	78.63	0.271	–[4]	–
Our approach	11.12	0.976	9.70	0.984	11.42	0.981	67.45	0.718

[1] `https://sites.google.com/site/andrealancichinetti/`
[2] `https://github.com/premgopalan/svinet`
[3] No results after more than 6 days of execution (> 8700 minutes).
[4] Presented out-of-memory problems and was terminated by the operating system.

coherent and global community structure for large networks. The approach is successful in several experiments with synthetic and large real-world networks.

References

1. Andersen, R., Chung, F., Lang, K.: Local graph partitioning using PageRank vectors. In: FOCS (2006)
2. Blondel, V., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. J. Stat. Mech. 2008(10), P10008+ (2008)
3. Campigotto, R., Guillaume, J.L., Seifi, M.: The power of consensus: Random graphs have no communities. In: ASONAM (2013)
4. Clauset, A.: Finding local community structure in networks. Phys. Rev. E 72, 026132 (2005)
5. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. Proc. Natl. Acad. Sci. U.S.A. 99(12), 7821–7826 (2002)
6. Gopalan, P., Blei, D.: Efficient discovery of overlapping communities in massive networks. Proc. Natl. Acad. Sci. U.S.A. 110(36), 14534–14539 (2013)
7. Lancichinetti, A., Fortunato, S.: Limits of modularity maximization in community detection. Phys. Rev. E 84(6), 066122+ (2011)
8. Lancichinetti, A., Fortunato, S.: Consensus clustering in complex networks. Sci. Rep. 2 (2012)
9. Lancichinetti, A., Fortunato, S., Kertész, J.: Detecting the overlapping and hierarchical community structure in complex networks. New J. of Phys. 11(3), 033015 (2009)
10. Lancichinetti, A., Fortunato, S., Radicchi, F.: Benchmark graphs for testing community detection algorithms. Phys. Rev, E 78(4) (2008)
11. Lancichinetti, A., Radicchi, F., Ramasco, J., Fortunato, S.: Finding statistically significant communities in networks. PLoS ONE 6(4), e18961+ (2011)

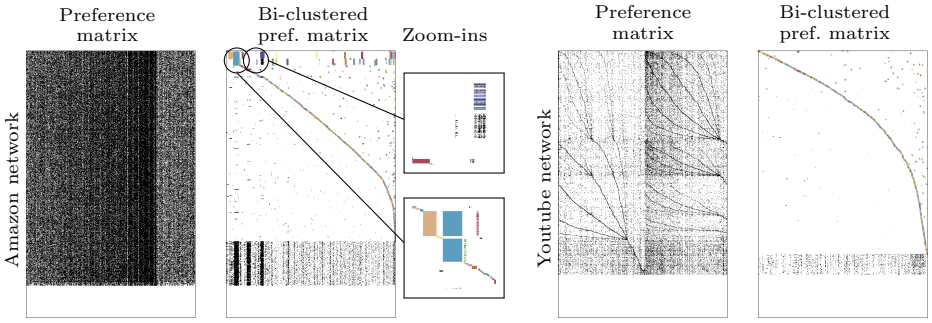

Fig. 1: Examples of bi-clustering the preference matrix for two real-world networks using the proposed approach. We permute rows and columns of the bi-clustered preference matrix to ease its visualization. The structural simplification obtained with our consensus approach is clear in the block-diagonal (plus overlaps) structure of the preference matrix.

12. Lee, M., Shen, H., Huang, J.Z., Marron, J.S.: Biclustering via sparse singular value decomposition. Biometrics 66(4), 1087–1095 (2010)
13. Li, T., Ding, C., Jordan, M.I.: Solving consensus and semi-supervised clustering problems using nonnegative matrix factorization. In: ICDM (2007)
14. Liu, C., Yang, H., Fan, J., He, L., Wang, Y.: Distributed nonnegative matrix factorization for web-scale dyadic data analysis on mapreduce. In: WWW (2010)
15. Newman, M.E.J., Girvan, M.: Finding and evaluating community structure in networks. Phys. Rev. E 69(026113) (2004)
16. Papalexakis, E.E., Sidiropoulos, N.D., Bro, R.: From K-Means to Higher-Way Co-Clustering: Multilinear decomposition with sparse latent factors. IEEE Trans. Signal Process. 61(2), 493–506 (2013)
17. Raghavan, U.N., Albert, R., Kumara, S.: Near linear time algorithm to detect community structures in large-scale networks. Phys. Rev. E 76, 036106 (2007)
18. Ramírez, I., Tepper, M.: Bi-clustering via MDL-based matrix factorization. In: CIARP (2013)
19. Rosvall, M., Bergstrom, C.: Maps of random walks on complex networks reveal community structure. Proc. Natl. Acad. Sci. U.S.A. 105(4), 1118–1123 (2008)
20. Tepper, M., Sapiro, G.: A bi-clustering framework for consensus problems. SIAM J. Imaging Sci. 7(4), 2488–2525 (2014)
21. Vega-Pons, S., Ruiz-Shulcloper, J.: A survey of clustering ensemble algorithms. Int. J. of Pattern Recognit. Artif. Intell. 25(03), 337–372 (2011)
22. Wang, L., Lou, T., Tang, J., Hopcroft, J.: Detecting community kernels in large social networks. In: ICDM (2011)
23. Witten, D.M., Tibshirani, R., Hastie, T.: A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. Biostatistics 10(3), 515–534 (2009)
24. Yang, J., Leskovec, J.: Defining and evaluating network communities based on ground-truth. In: MDS (2012)