

Spectral Triadic Decompositions of Real-World Networks

Sabyasachi Basu* Suman Kalyan Bera† C. Seshadhri‡

Abstract

A fundamental problem in mathematics and network analysis is to find conditions under which a graph can be partitioned into smaller pieces. The most important tool for this partitioning is the Fiedler vector or discrete Cheeger inequality. These results relate the graph spectrum (eigenvalues of the normalized adjacency matrix) to the ability to break a graph into two pieces, with few edge deletions. An entire subfield of mathematics, called spectral graph theory, has emerged from these results. Yet these results do not say anything about the rich community structure exhibited by real-world networks, which typically have a significant fraction of edges contained in numerous densely clustered blocks. Inspired by the properties of real-world networks, we discover a new spectral condition that relates eigenvalue powers to a network decomposition into densely clustered blocks. We call this the *spectral triadic decomposition*. Our relationship exactly predicts the existence of community structure, as commonly seen in real networked data. Our proof provides an efficient algorithm to produce the spectral triadic decomposition. We observe on numerous social, coauthorship, and citation network datasets that these decompositions have significant correlation with semantically meaningful communities.

1 Introduction

The existence of clusters or community structure is one of the most fundamental properties of real-world networks. Across various scientific disciplines, be it biology, social sciences, or physics, the modern study of networks has often deal with the community structure of these data. Procedures that discover community structure have formed an integral part of network science algorithmics. Despite the large variety of formal definitions of a community in a network, there is broad agreement that it constitutes a dense substructure in an overall sparse network. Indeed, the discover of local density (also called *clustering coefficients*) goes back to the birth of network science.

Even beyond network science, graph partitioning is a central problem in applied mathematics and the theory of algorithms. Determining when such a partitioning is possible is a fundamental question that one straddles graph theory, harmonic analysis, differential geometry, and theoretical computer science. There is large body of mathematical and scientific research on how to break up a graph into smaller pieces.

Arguably, the most important mathematical tool for this partitioning problem is the *discrete Cheeger inequality* or the *Fiedler vector*. This result is the cornerstone of spectral graph theory and relates the eigenvalues of the graph Laplacian to the combinatorial structure. Consider an undirected graph $G = (V, E)$ with n vertices. Let d_i denote the degree of the vertex i . The *normalized adjacency matrix*, denoted $\mathcal{A}$, is the $n \times n$ matrix where the entry $\mathcal{A}_{ij}$ is $1/\sqrt{d_i d_j}$ if (i, j) is an edge, and zero otherwise. (All diagonal entries are zero.) One can think of this entry as the “weight” of the edge between i and j .

Let $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$ denote the n eigenvalues of the non-negative symmetric matrix $\mathcal{A}$. The largest eigenvalue λ_1 is always one. A basic fact is that $\lambda_2 = 1$ iff G is disconnected. The discrete Cheeger inequality proves that if λ_2 is close to 1 (has value $\geq 1 - \varepsilon$), then G is “close” to being disconnected. Formally, there

*Department of Computer Science and Engineering, University of California, Santa Cruz sbasu3@ucsc.edu

†Katana Graph sumankalyanbera@gmail.com

‡Department of Computer Science and Engineering, University of California, Santa Cruz sesh@ucsc.edu

SB and CS are supported by NSF DMS-2023495, CCF-1740850, 1839317, 1813165, 1908384, 1909790, and ARO Award W911NF1910294.

exists a set S of vertices that can be disconnected (from the rest of G) by removing an $O(\sqrt{\varepsilon})$ -fraction of edges incident to S . The set of edges removed is called a low conductance cut.

We can summarize these observations as:

Basic fact: Spectral gap is zero $\implies G$ is disconnected

Cheeger bound: Spectral gap is close to zero $\implies G$ can be disconnected by low conductance set

The quantitative bound is one of the most important results in the study of graphs and network analysis. There is a rich literature of generalizing this bound for higher-order networks and simplicial complexes. We note that many modern algorithms for finding communities in real-world networks are based on the Cheeger inequality in some form. The seminal Personalized PageRank algorithm provides a local version of the Cheeger bound.

For modern network analysis and community structure, there are several unsatisfying aspects of the Cheeger inequality. Despite the variety of formal definitions of a community in a network, there is broad agreement that it constitutes many densely clustered substructures in an overall sparse network. The Cheeger inequality only talks of disconnecting G into two parts. Even generalizations of the Cheeger inequality only work for a constant number of parts [17]. Real-world networks decompose into an extremely large number of blocks/communities, and this number often scales with the network size [18, 27]. Secondly, the Cheeger bound works when the spectral gap is close to zero, which is often not true for real-world networks [18]. Real-world networks possess the small-world property [16]. But this property implies large spectral gap. Thirdly, Cheeger-type inequalities make no assertion on the interior of parts obtained. In community structure, we typically expect the interior to be dense and potentially assortative (possessing vertices of similar degree).

The main question that we address: *is there a spectral quantity that predicts the existence of real-world community structure?*

1.1 Main result

We take inspiration from a central property of real-world graphs, the abundance of triangles [39, 27]. This abundance is widely seen across graphs that come by disparate domains. Recent work in network science and data mining have used the triangles to effectively cluster graphs. There is much evidence that the triangle structure aids finding communities in graphs [26, 36, 3, 37].

In network science, the triangle count is often expressed in terms of the *transitivity* or global clustering coefficient [8, 38]. We define the *spectral transitivity* of the graph G .

Definition 1.1. *The spectral transitivity of G , denoted $\tau(G)$, is defined as follows¹. (Recall that the λ_i s are the eigenvalues of the normalized adjacency matrix.)*

$$\tau(G) = \frac{\sum_{i \leq n} \lambda_i^3}{\sum_{i \leq n} \lambda_i^2}. \quad (1)$$

Standard arguments show that the spectral transitivity is a degree weighted transitivity. The numerator is a weighted sum over all triangles, while the denominator (squared Frobenius norm) is a weighted sum over edges (Lemma 3.5).

Observe that since $\lambda_i \leq 1$, $\tau \leq 1$. When τ reaches its maximum value of $1 - 1/(n - 1)$, one can show that G is a clique (Lemma 3.6). We formalize the notion of "clique-like" submatrices through the concept of uniformity. For a symmetric matrix M and a subset S of its columns/rows, we use $M|_S$ to denote the square submatrix restricted to S (on both columns and rows).

Definition 1.2. *Let $\alpha \in (0, 1]$. Let $\mathcal{A}$ be the normalized adjacency matrix of a graph G . For any subset of vertices S , $|\mathcal{A}|_S$ is called α -strongly uniform if at least an α -fraction of non diagonal entries have values in the range $[\alpha/(|S| - 1), 1/\alpha(|S| - 1)]$.*

For $s \in S$, let $N(s, S)$ denote the neighborhood of s in S (we define edges by non-zero entries). An α -uniform matrix is strongly α -uniform if for at least an α -fraction of $s \in S$, $\mathcal{A}|_{N(s, S)}$ is also α -uniform.

¹If G (or the normalized adjacency matrix $\mathcal{A}$) are obvious from context, we simply refer to τ instead of $\tau(G)$.

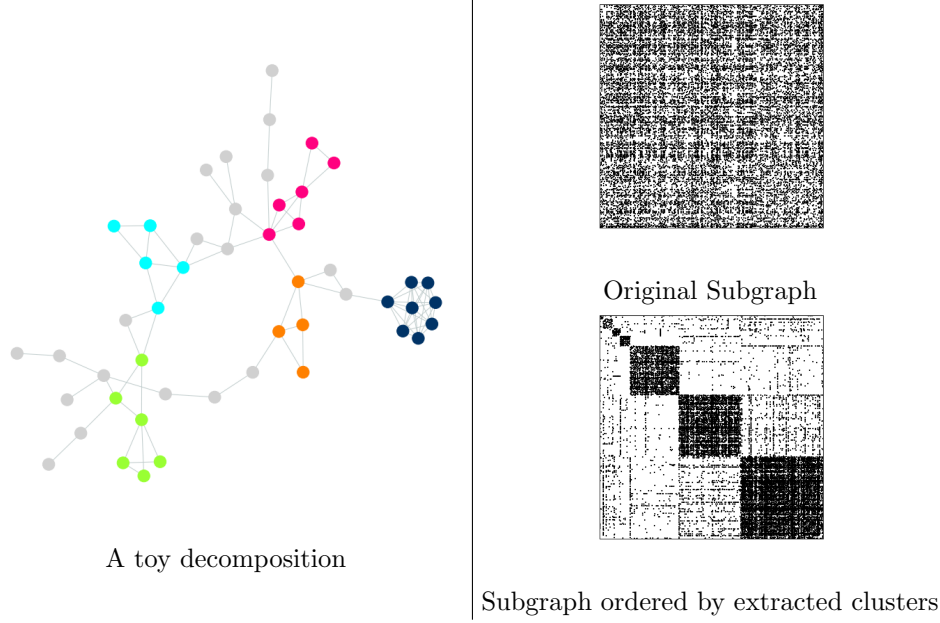


Figure 1: On the left, as a small example, we consider a subgraph induced by 155 vertices and $\tau = 0.49$ from a coauthorship network of Condensed Matter Physics researchers[21], and show a spectral triadic decomposition of the largest connected component, which has 49 vertices. Each cluster is colored differently. We see how each cluster forms a densely connected component within an otherwise sparse graph. Also note that the clusters vary in size. The gray vertices do not participate in the decomposition, since they do not add significant to the cluster structure. On the right, we look at the adjacency matrices pre and post decomposition. The top figure is a spy plot of the adjacency matrix of 488 connected vertices from a Facebook network ([35],[34]) taken from the network repository[25], a graph with $\tau = 0.122$. As a demonstration, we compute the spectral triadic decomposition of this subnetwork. We group the columns/rows by the clusters in the spy plot on the bottom. The latent community structure is immediately visible. Note that there exists many such blocks of varying sizes.

Observe that the normalized adjacency matrix of a clique is (strongly) 1-uniform. But submatrices of this matrix are not. Roughly speaking, a constant uniform submatrix corresponds to a dense subgraph of (say) size k where the *total* degrees of vertices is $\Theta(k)$. Strong uniformity is closely related to *clustering coefficients*, which is the edge density of neighborhoods. It is well-known that real-world graphs have high clustering coefficients [39, 27]. A strongly uniform submatrix essentially exhibits high clustering coefficients.

Our main theorem states that any graph with constant spectral transitivity can be decomposed into constant uniform blocks. We use $\|M\|_2$ to denote the Frobenius norm of matrix M .

Theorem 1.3 (Spectral Theorem). *There exist absolute constants $\delta > 0$ and $c > 0$ such that the following holds. Let $\mathcal{A}$ be the normalized adjacency matrix of a graph with spectral transitivity τ .*

There exists a collection of disjoint sets of vertices $X_1, X_2, \dots, X_k$ satisfying the following conditions:

1. (Cluster structure) *For all $i \leq k$, $\mathcal{A}|_{X_i}$ is strongly $\delta\tau^c$ -uniform.*
2. (Coverage) $\sum_{i \leq k} \|\mathcal{A}|_{X_i}\|_2^2 \geq \delta\tau^c \|\mathcal{A}\|_2^2$.

We call this output the *spectral triadic decomposition*. Our proof also yields an efficient algorithm that computes the decomposition, whose running time is dominated by a triangle enumeration. Details are given in Theorem 6.1 and §6.

1.2 Significance of Theorem 1.3

One can think of Theorem 1.3 as a type of Cheeger inequality that is relevant to the structure of real-world social networks. We explain how it captures many of the salient properties of clusters in real-world networks. In this discussion, we will assume that τ is a constant.

The spectral transitivity: We find it remarkable that a bound on a single spectral quantity, τ , implies such a rich decomposition. The spectral transitivity τ captures a key property of real-world graphs, the abundance of triangles. While there is a rich body of empirical work on using triangles to cluster graphs, there is no theory explaining *why* triangles are so useful. Theorem 1.3 gives a spectral-theoretic explanation.

The spectral transitivity is a weighted version of the transitivity, which is typically around 0.1 for real-world graphs². We also note that the final algorithm that computes the decomposition focuses on triangle cuts, which is a popular empirical technique for finding clusters in social networks [3, 37].

The strong uniformity of clusters: Each cluster X_i of the spectral triadic decomposition is (constant) strongly uniform. While there is no one definition of a "community" in real-world graphs, the definition of strong uniformity captures many basic concepts. Most importantly, X_i is internally dense in edges. Let $|X_i| = k$. Then $\Omega(k^2)$ entries in X_i are $\Omega(1/k)$, which (by averaging) implies that a constant fraction of X_i involves vertices of degree $\Theta(k)$. Thus, a constant fraction of X_i vertices have a constant fraction of their neighbors in X_i . Moreover, the submatrix of every neighborhood in X_i is also uniform. This is quite consistent with the typical notion of a social network community.

Crucially, Theorem 1.3 gives a condition on the *internal* structure of the decomposition. This addresses a key weakness of the Cheeger inequality.

The coverage condition: It is natural to measure the "mass" of a matrix by the squared Frobenius norm. The clusters of spectral triadic decomposition of Theorem 1.3 capture a constant fraction of this squared norm. This is consistent with the fact that a constant fraction of the edges in a real-world graph are *not* community edges [20, 12, 16, 27]. Any decomposition into communities would avoid these "long-range" edges, excluding a constant fraction of the matrix mass.

Robustness to noise: Taking the above point further, the non-community edges are often modeled as stochastic (or noisy). The underlying cluster structure of a real-world graph is robust to such perturbations. Adding (say) an Erdős-Rényi graph with $\Theta(n)$ edges can only affect the spectral transitivity by a constant factor (by changing the Frobenius norm). Theorem 1.3 would only be affected by constant factors. Note that the spectral gap, on the other hand, can dramatically increase by such noise.

Spectral graph theory inspired by real-world graphs: We consider Theorem 1.3 as opening up a new direction in spectral graph theory. At a mathematical level, Theorem 1.3 is like a Cheeger inequality,

²Our experiments on these real-world graphs yield similar values for the spectral transitivity.

where a spectral condition implies a graph theoretic property. But all aspects of [Theorem 1.3](#) (the notion of spectral transitivity and the properties of the decomposition) are inspired by the observed properties of real-world graphs.

1.3 A comparison with Gupta, Roughgarden and Seshadhri’s result

Our algorithm is heavily influenced by the problem studied by Gupta, Roughgarden and Seshadhri [13], who sought a description for real world social networks that did not depend on generative models. We hereafter refer to this result as GRS. The objective in their work was to find a combinatorial assumption that holds in every common model; but strong enough to imply sufficient structure, and allow algorithms to be devised. They define the triangle density of a graph, which we shall call $t_d(G)$ for a graph G .

Definition 1.4 (Triangle Density, from GRS). *The triangle density of an undirected graph $G = (V, E)$ is defined as $t_d(G) = 3t(G)/w(G)$. Here, $t(G)$ refers to the number of triangles in the graph, and $w(G)$ refers to the number of wedges, or two hop paths in the graph.*

The triangle density is also often referred to as the (combinatorial) transitivity, but we use the former term to avoid confusion with spectral transitivity. It is useful to think of this as a measure of rate of closure in triples of connected vertices. In this setup, the main theorem of GRS guarantees that if you have a graph G such that $t_d(G) = \varepsilon$, then it admits a decomposition into a *tightly knit family* such that each subgraph has triangle and edge density ε , and the family contains an ε fraction of $t(G)$ (no guarantees are given on the edges preserved).

Consider a graph G on n vertices with an apex vertex v that has degree $n - 1$, and a series of disjoint subgraphs $G_1, \dots, G_{n/\log n}$, each with $\log n$ vertices. Assume that each subgraph is a complete graph in itself. It is easy to see that:

1. The number of wedges is $\Theta(n^2)$.
2. The number of triangles is $\Theta(n \log n)$.
3. $t_d(G)$ is $o(1)$, as most of the wedges that v participates in (of the form (u, v, w) for $u, w \in V(G)$) do not close to form triangles.

In this case, where the triangle density is vanishingly small, GRS fails to provide any guarantees. We emphasize that this is a very relevant occurrence in real world social networks: consider a celebrity who is followed by large numbers of people spread across different dense communities.

Our crucial technical contribution is finding the right degree based weighting to fix this issue. We downweight vertices of high degree such that we can still have a natural quantity that represents this idea and also gives us strong guarantees algorithmically. Consequently, observe that the spectral transitivity is high for this graph. This reweighting provides objectively stronger guarantees, and allows our results to hold for more general classes of graphs.

Our reweighting procedure brings with it its own set of challenges. The analysis in GRS depends heavily on dealing with Jaccard similarity, which is a real in $[0, 1]$. For the local analysis, we deal with an analogous ratio that is not quite as well behaved and can take arbitrarily large values. Consequently, the analysis is fairly more complicated to give guarantees in terms of uniformity, and our algorithm must be a lot more careful in picking vertices to include in a cluster. Moreover, our results are far more universal, because we state results in terms of matrix norms.

2 Related Work

Spectral graph theory is a deep field of study with much advancement over the past two decades. We refer the readers to the classic textbook by Chung [7], and the tutorial [29] and lecture notes [28] by Spielman.

The cluster structure of real-world networks has attracted attention from the early days of network science [10, 22]. Fortunato’s (somewhat dated) survey on community detection has details of the key results [9]. There is no definitive model for social networks, but it is generally accepted that they have many dense clusters with sparse connections between them [6, 18, 27]. The study of triangles and neighborhood density

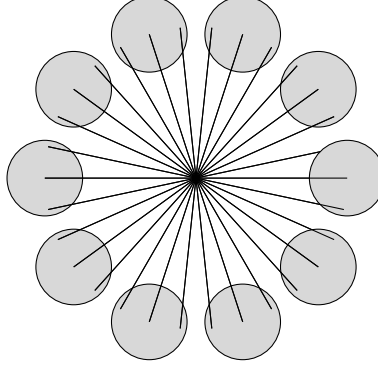


Figure 2: An example of a graph that GRS fails to tackle. Each shaded circle is a complete subgraph of size $\log n$, and the apex vertex is connected to every other vertex. There are no edges in between the clusters. This graph has low triangle density, but high spectral transitivity.

goes back to the early days of social science theory [14, 15, 5, 8]. Early network science papers popularized the notion of clustering coefficients and transitivity as useful measures [39]. The use of triangles to find such clusters is a more recent development in network science. A number of contemporary results explicit use triangle information for algorithmic purposes [26, 36, 3, 37]. Our main theorem is inspired by these applications.

While the Cheeger inequality by itself is not useful for real-world graph clustering, local versions of spectral clustering are extremely useful [30, 2]. We stress that these results do not relate the graph spectrum to the partitions. But the algorithm is inspired by the proof of the Cheeger inequality. Many results on the cluster structure of real-world graphs [18, 11] use the Personalized PageRank method [2]. Some local partitioning methods yield bounds on the internal structure of clusters [17, 23, 24].

Most relevant to our work is the result of Gupta, Roughgarden, and Seshadhri [13]. They prove a decomposition theorem for triangle-rich graphs, as measured by graph transitivity. Their main result shows that a triangle-dense graph can be clustered into dense clusters. The results of [13] do not have any spectral connection, nor do they provide the kind of uniformity or coverage bounds of Theorem 1.3. Our main insight is in generalizations of their proof technique, which leads to connections with graph spectrum. We adapt the [13] proof to deal with normalized adjacency matrix, which adds many complications because of the non-uniformity of entries.

3 Preliminaries

We use V, E, T to denote the sets of vertices, edges, and triangles of G , respectively. For any subgraph H of G , we use V_H, E_H, T_H to denote the corresponding sets within H . For any edge e , let $T_H(e)$ denote the set of triangles in H containing e .

For any vertex v , let d_v denote the degree of v (in G).

We first define the notion of *weights* for edges and triangles. We will think of edges and triangles as unordered sets of vertices.

Definition 3.1. For any edge $e = (u, v)$, define the weight $\text{wt}(e)$ to be $\frac{1}{d_u d_v}$. For any triangle $t = (u, v, w)$, define the weight $\text{wt}(t)$ to be $\frac{1}{d_u d_v d_w}$.

For any set S consisting solely of edges or triangles, define $\text{wt}(S) = \sum_{s \in S} \text{wt}(s)$.

We state some basic facts that relate the sum of weights to sum of eigenvalue powers. Let $S \subset V$ be any subset of vertices, and let $\mathcal{A}|_S$ denote the submatrix of $\mathcal{A}$ restricted to S . We use $\lambda_i(S)$ to denote the

i th largest eigenvalue of the symmetric submatrix $\mathcal{A}|_S$. Abusing notation, we use E_S and T_S to denote the edges and triangles contained in the graph induced on S .

Claim 3.2. $\sum_{i \leq |S|} \lambda^2(S)_i = 2 \sum_{e \in E(S)} \text{wt}(e)$

Proof. By the properties of the Frobenius norm of matrices, $\sum_{i \leq |S|} \lambda_i^2 = \sum_{s,t \in S} \mathcal{A}_{st}^2$. Note that $\mathcal{A}_{st} = A_{st}/\sqrt{d_s d_t}$. Hence, $\sum_{s,t} \mathcal{A}_{st}^2 = 2 \sum_{e=(u,v) \in E(S)} 1/d_u d_v$. (We get a 2-factor because each edge (u, v) appears twice in the adjacency matrix.) $\square$

Claim 3.3. $\sum_{i \leq |S|} \lambda^3(S)_i = 6 \sum_{t \in T(S)} \text{wt}(t)$.

Proof. Note that $\sum_{i \leq |S|} \lambda^3(S)_i$ is the trace of $(\mathcal{A}|_S)^3$. The diagonal entry $(\mathcal{A}|_S)_{ii}^3$ is precisely $\sum_{s \in S} \sum_{s' \in S} \mathcal{A}_{is} \mathcal{A}_{ss'} \mathcal{A}_{s'i}$. Note that $\mathcal{A}_{is} \mathcal{A}_{ss'} \mathcal{A}_{s'i}$ is non-zero iff (i, s, s') form a triangle. In that case, $\mathcal{A}_{is} \mathcal{A}_{ss'} \mathcal{A}_{s'i} = 1/\sqrt{d_i d_s} \cdot 1/\sqrt{d_s d_{s'}} \cdot 1/\sqrt{d_{s'} d_i} = \text{wt}((i, s, s'))$. We conclude that $(\mathcal{A}|_S)_{ii}^3$ is $2 \sum_{t \in T(S), t \ni i} \text{wt}(t)$. (There is a 2 factor because every triangle is counted twice.)

Thus, $\sum_{i \leq n} \lambda^3(S)_i = \sum_i 2 \sum_{t \in T, t \ni i} \text{wt}(t) = 2 \sum_{t \in T} \sum_{i \in t} \text{wt}(t) = 6 \sum_{t \in T} \text{wt}(t)$. (The final 3 factor appears because a triangle contains exactly 3 vertices.) $\square$

Claim 3.4. $\sum_{t \in T(S)} \text{wt}(t) \leq \|\mathcal{A}|_S\|_2^2/6$.

Proof. By Claim 3.3 $\sum_{t \in T(S)} \text{wt}(t) = \sum_{i \leq |S|} \lambda^3(S)_i/6$. The maximum eigenvalue of $\mathcal{A}$ is 1, and since $\mathcal{A}|_S$ is a submatrix, $\lambda(S)_1 \leq 1$ (Cauchy's interlacing theorem). Thus, $\sum_{i \leq |S|} \lambda^3(S)_i \leq \sum_{i \in |S|} \lambda^2(S)_i = \|\mathcal{A}|_S\|_2^2$. $\square$

As a direct consequence of the previous claims applied on $\mathcal{A}$, we get the following characterization of the spectral triadic content in terms of the weights.

Lemma 3.5. $\tau = \frac{3 \sum_{t \in T} \text{wt}(t)}{\sum_{e \in E} \text{wt}(e)}$.

While the following bound is not necessary for our main result, it is instructive to see the largest possible value of the spectral transitivity.

Lemma 3.6. Consider normalized adjacency matrices $\mathcal{A}$ with n vertices. The maximum value of $\tau(\mathcal{A})$ is $1 - 1/(n-1)$. This value is attained for the unique strongly 1-uniform matrix, the normalized adjacency matrix of the n -clique.

Proof. First, consider the normalized adjacency matrix $\mathcal{A}$ of the n -clique. All off-diagonal entries are precisely $1/(n-1)$ and $\mathcal{A}$ can be expressed as $(n-1)^{-1}(\mathbf{1}\mathbf{1}^T - I)$. The matrix $\mathcal{A}$ is 1-regular. The largest eigenvalue is 1 and all the remaining eigenvalues are $-1/(n-1)$. Hence, $\sum_i \lambda_i^3 = 1 - (n-1)/(n-1)^3 = 1 - 1/(n-1)^2$. The sum of squares of eigenvalue is $\sum_i \lambda_i^2 = 1 + (n-1)/(n-1)^2 = 1 + 1/(n-1)$. Dividing,

$$\frac{\sum_{i \leq n} \lambda_i^3}{\sum_{i \leq n} \lambda_i^2} = 1 - 1/(n-1).$$

Since the matrix has zero diagonal, the trace $\sum_i \lambda_i$ is zero. We will now prove the following claim.

Claim 3.7. Consider any sequence of numbers $1 = \lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$ such that $\forall i, |\lambda_i| \leq 1$ and $\sum_i \lambda_i = 0$. If $\sum_i \lambda_i^3 \geq (1 - 1/(n-1)) \sum_i \lambda_i^2$, then $\forall i > 1, \lambda_i = -1/(n-1)$.

Proof. Let us begin with some basic manipulations.

$$\sum_i \lambda_i^3 \geq [1 - 1/(n-1)] \sum_i \lambda_i^2 \tag{2}$$

$$\begin{aligned} \implies 1 + \sum_{i>1} \lambda_i^3 &\geq [1 - 1/(n-1)] \cdot (1 + \sum_{i>1} \lambda_i^2) \\ \implies \sum_{i>1} \lambda_i^3 &\geq [1 - 1/(n-1)] \sum_{i>1} \lambda_i^2 - 1/(n-1). \end{aligned} \tag{3}$$

For $i > 1$, define $\delta_i := \lambda_i + 1/(n-1)$. Note that $\sum_{i>1} \lambda_i = -1$, so $\sum_{i>1} \delta_i = 0$. Moreover, $\forall i > 1$, $\delta_i \leq 1 + 1/(n-1)$. We plug in $\lambda_i = \delta_i - 1/(n-1)$ in (3).

$$\begin{aligned} \sum_{i>1} \left[\delta_i - 1/(n-1) \right]^3 &\geq [1 - 1/(n-1)] \sum_{i>1} \left[\delta_i - 1/(n-1) \right]^2 - 1/(n-1) \\ \implies \sum_{i>1} \left[\delta_i^3 - 3\delta_i^2/(n-1) + 3\delta_i/(n-1)^2 - 1/(n-1)^3 \right] \\ &\geq [1 - 1/(n-1)] \sum_{i>1} \left[\delta_i^2 - 2\delta_i/(n-1) + 1/(n-1)^2 \right] - 1/(n-1). \end{aligned}$$

Recall that $\sum_{i>1} \delta_i = 0$. Hence, we can simplify the above inequality.

$$\begin{aligned} \sum_{i>1} \delta_i^3 - (3/(n-1)) \sum_{i>1} \delta_i^2 - 1/(n-1)^2 \\ \geq [1 - 1/(n-1)] \sum_{i>1} \delta_i^2 + 1/(n-1) - 1/(n-1)^2 - 1/(n-1) \\ \implies \sum_{i>1} \delta_i^3 \geq [1 + 2/(n-1)] \sum_{i>1} \delta_i^2. \quad (\text{Canceling terms and rearranging}) \end{aligned}$$

Since $\delta_i \leq (1 + 1/(n-1))$, we get that $\sum_{i>1} \delta_i^3 \leq [1 + 1/(n-1)] \sum_{i>1} \delta_i^2$. Combining with the above inequality, we deduce that $[1 + 2/(n-1)] \sum_{i>1} \delta_i^2 \leq [1 + 1/(n-1)] \sum_{i>1} \delta_i^2$. This can only happen if $\sum_{i>1} \delta_i^2$ is zero, implying all δ_i values are zero. Hence, for all $i > 1$, $\lambda_i = -1/(n-1)$. $\square$

With this claim, we conclude that any matrix $\mathcal{A}$ maximizing the ratio of cubes and squares of eigenvalues has a fixed spectrum. It remains to prove that a unique normalized adjacency matrix has this spectrum. We use the rotational invariance of the Frobenius norm: sum of squares of entries of $\mathcal{A}$ is the same as the sum of squares of eigenvalues. Thus,

$$\sum_{(u,v) \in E} \frac{2}{d_u d_v} = 1 + \frac{1}{n-1} = \frac{n}{n-1}. \quad (4)$$

Observe that $\frac{2}{d_u d_v} \geq 1/(d_u(n-1)) + 1/(d_v(n-1))$, since all degrees are at most $n-1$. Summing this inequality over all edges,

$$\sum_{(u,v) \in E} \frac{2}{d_u d_v} \geq \sum_{v \in V} \sum_{u \in N(v)} \frac{1}{d_v(n-1)} = \sum_{v \in V} \frac{d_v}{d_v(n-1)} = \frac{n}{n-1}. \quad (5)$$

Hence, for (4) to hold, for all edges (u, v) , we must have the equality $\frac{2}{d_u d_v} = 1/(d_u(n-1)) + 1/(d_v(n-1))$. That implies that for all edge (u, v) , $d_u = d_v = n-1$. So all vertices have degree $(n-1)$, and the graph is an n -clique. $\square$

We will need the following “reverse Markov inequality” for some intermediate proofs.

Lemma 3.8. *Consider a random variable Z taking values in $[0, b]$. If $\mathbf{E}[Z] \geq \sigma b$, then $\Pr[Z \geq \sigma b/2] \geq \sigma/2$.*

Proof. In the following calculations, we will upper bound the conditional expectation by the maximum value (under that condition).

$$\sigma b \leq \mathbf{E}[Z] = \Pr[Z \geq \sigma b/2] \cdot \mathbf{E}[Z|Z \geq \sigma b/2] + \Pr[Z < \sigma b/2] \cdot \mathbf{E}[Z|Z < \sigma b/2] \quad (6)$$

$$\leq \sigma b/2 \cdot \mathbf{E}[Z|Z \leq \sigma b/2] \quad (7)$$

$$\leq \Pr[Z \geq \sigma b/2] \cdot b + \sigma b/2 \quad (8)$$

We rearrange to complete the proof. $\square$

4 Cleaned graphs and extraction

For convenience, we set $\varepsilon = \tau/6$.

Definition 4.1. A connected subgraph H is called *clean* if $\forall e \in E(H)$, $\text{wt}(T_H(e)) \geq \varepsilon \text{wt}(e)$.

Algorithm 1 Extract(H)

- 1: Pick $v \in V(H)$ that minimizes d_v .
 - 2: Construct the set $L := \{u \mid (u, v) \in E(H), d_u \leq 2\varepsilon^{-1}d_v\}$ (L is the set of low degree neighbors of v in H .)
 - 3: For every vertex $w \in V(H)$, define ρ_w to be the total weight of triangles of the form (w, u, u') where $u, u' \in L$.
 - 4: Sort the vertices in decreasing order of ρ_w , and construct the “sweep cut” C to be the smallest set satisfying $\sum_{w \in C} \rho_w \geq (1/2) \sum_{w \in V(H)} \rho_w$.
 - 5: Output $X := \{v\} \cup L \cup C$.
-

The main theorem of this section follows.

Theorem 4.2. Suppose the subgraph H is connected and clean. Let X denote the output of the procedure **Extract**(H). Then

$$\sum_{t \in T(H), t \subseteq X} \text{wt}(t) \geq (\varepsilon^8/2000) \sum_{t \in T(H), t \cap X \neq \emptyset} \text{wt}(t)$$

(The triangle weight contained inside X is a constant fraction of the triangle weight incident to X .)

Moreover, $\mathcal{A}|_X$ is strongly $\delta\varepsilon^{12}$ -uniform.

We will need numerous intermediate claims to prove this theorem. We use v , L , and C as defined in **Extract**(H). We use N to denote the neighborhood of v in H . Note that $L \subseteq N$.

For any vertex $u \in N$, we define the set of partners $P(u)$ to be $\{w : (u, v, w) \in T_H\}$.

The following lemma is an important tool in our analysis.

Lemma 4.3. For any $u \in N$, $\sum_{w \in P(u) \cap L} d_w^{-1} \geq \varepsilon/2$.

Proof. Let $e = (u, v)$. Since H is clean, $\text{wt}(T_H(e)) \geq \varepsilon \text{wt}(e)$. Expanding out the definition of weights,

$$\sum_{w: (u, v, w) \in T_H} \frac{1}{d_u d_v d_w} \geq \frac{\varepsilon}{d_u d_v} \implies \sum_{w \in P(u)} d_w^{-1} \geq \varepsilon. \quad (9)$$

Note that L (as constructed in **Extract**(H)) is the subset of N consisting of vertices with degree at most $2\varepsilon^{-1}d_v$. For $w \in N \setminus L$, we have the lower bound $d_w \geq 2\varepsilon^{-1}d_v$. Hence,

$$\sum_{w \in N \setminus L} d_w^{-1} \leq |N \setminus L|(\varepsilon/2)d_v^{-1} \leq d_v \times (\varepsilon/2)d_v^{-1} = \varepsilon/2. \quad (10)$$

In the calculation below, we split the sum of (9) into the contribution from L and from outside L . We apply (10) to bound the latter contribution.

$$\varepsilon \leq \sum_{w \in P(u)} d_w^{-1} \leq \sum_{w \in P(u) \cap L} d_w^{-1} + \sum_{w \in N \setminus L} d_w^{-1} \leq \sum_{w \in P(u) \cap L} d_w^{-1} + \varepsilon/2. \quad (11)$$

□

Claim 4.4. $|L| \geq \varepsilon d_v/2$

Proof. Since H is connected, there must exist some edge $e = (u, v) \in E(H)$. By [Lemma 4.3](#), $\sum_{w \in P(u) \cap L} d_w^{-1} \geq \varepsilon/2$. Hence, $\sum_{w \in L} d_w^{-1} \geq \varepsilon/2$. Since v is the vertex in $V(H)$ minimizing d_v , for any $w \in V(H)$, $d_w \geq d_v$. Thus,

$$\varepsilon/2 \leq \sum_{w \in L} d_w^{-1} \leq \sum_{w \in L} d_v^{-1} = |L|d_v^{-1}. \quad (12)$$

□

Claim 4.5. $\sum_{e \in E(H), e \subseteq L} \text{wt}(e) \geq \varepsilon^2/8$.

Proof. By [Lemma 4.3](#), $\forall w \in L$, $\sum_{w' \in P(w) \cap L} d_{w'}^{-1} \geq \varepsilon/2$. We multiply both sides by d_w^{-1} and sum over all $w \in L$.

$$\sum_{w \in L} \sum_{w' \in P(w) \cap L} (d_w d_{w'})^{-1} \geq (\varepsilon/2) \sum_{w' \in L} d_{w'}^{-1}. \quad (13)$$

By [Lemma 4.3](#), $\sum_{w' \in L} d_{w'}^{-1} \geq \varepsilon/2$. Note that $w' \in P(w)$ only if $(w, w') \in E(H)$. Hence,

$\sum_{w \in L} \sum_{w' \in L, (w, w') \in E(H)} \text{wt}((w, w')) \geq \varepsilon^2/4$. Note that the summation counts all edges twice, so we divide by 2 to complete the proof. □

We now come to the central calculations of the main proof. Recall, from the description of **Extract**, that ρ_w is the total triangle weight of the triangles (w, u, u') , where $u, u' \in L$. We will prove that $\sum_w \rho_w$ is large; moreover, there are a few entries that dominate the sum. The latter bound is crucial to arguing that the sweep set C is not too large.

Claim 4.6. $\sum_{w \in V(H)} \rho_w \geq \varepsilon^3/8$.

Proof. Note that $\sum_{w \in V(H)} \rho_w$ is equal to $\sum_{e \in E(H), e \subseteq L} \text{wt}(T_H(e))$. Both these expressions give the total weight of all triangles in H that involve two vertices in L . Since H is clean, for all edges $e \in E(H)$, $\text{wt}(T_H(e)) \geq \varepsilon \text{wt}(e)$. Hence, $\sum_{e \in E(H), e \subseteq L} \text{wt}(T_H(e)) \geq \varepsilon \sum_{e \in E(H), e \subseteq L} \text{wt}(e)$. Applying [Claim 4.5](#), we can lower bound the latter by $\varepsilon^3/8$. □

We now show that a few ρ_w values dominate the sum, using a somewhat roundabout argument. We upper bound the sum of square roots.

Claim 4.7. $\sum_{w \in V(H)} \sqrt{\rho_w} \leq 2\varepsilon^{-1} \sqrt{d_v}$

Proof. Let c_w be the number of vertices in L that are neighbors (in H) of w . Note that for any triangle (u, u', w) where $u, u' \in L$, both u and u' are common neighbors of w and v . The number of triangles (u, u', w) where $u, u' \in L$ is at most c_w^2 . The weight of any triangle in H is at most d_v^{-3} , since d_v is the lowest degree (in G) of all vertices in H . As a result, we can upper bound $\rho_w \leq d_v^{-3} c_w^2$.

Taking square roots and summing over all vertices,

$$\sum_{w \in V(H)} \sqrt{\rho_w} \leq d_v^{-3/2} \sum_{w \in V(H)} c_w \quad (14)$$

Note that $\sum_{w \in V(H)} c_w$ is exactly the sum over $u \in L$ of the degrees of u in the subgraph H . (Every edge incident to $u \in L$ gives a unit contribution to the sum $\sum_{w \in V(H)} c_w$.) By definition, every vertex in L has degree in H at most $2\varepsilon^{-1}d_v$. The size of L is at most d_v .

Hence, $\sum_{w \in V(H)} c_w \leq 2\varepsilon^{-1}d_v^2$. Plugging into (14), we deduce that $\sum_{w \in V(H)} \sqrt{\rho_w} \leq 2\varepsilon^{-1} \sqrt{d_v}$. □

We now prove that the sweep cut C is small, which is critical to proving [Theorem 4.2](#).

Claim 4.8. $|C| \leq 144\varepsilon^{-5}d_v$.

277 *Proof.* For convenience, let us reindex vertices so that $\rho_1 \geq \rho_2 \geq \rho_3 \dots$. Let $r \leq n$ be an arbitrary index.
 278 Because we index in non-increasing order, note that $\sum_{j \leq n} \rho_j \geq r\rho_r$. Furthermore, $\forall j > r, \rho_j \leq \rho_r$.

$$\sum_{j>r} \rho_j \leq \sqrt{\rho_r} \sum_{j>r} \sqrt{\rho_j} \leq \sqrt{\frac{\sum_{j \leq n} \rho_j}{r}} \sum_{j \leq n} \sqrt{\rho_j} = \left[\frac{\sum_{j \leq n} \sqrt{\rho_j}}{\sqrt{r} \cdot \sqrt{\sum_{j \leq n} \rho_j}} \right] \sum_{j \leq n} \rho_j \quad (15)$$

Observe that [Claim 4.7](#) gives an upper bound on the numerator, while [Claim 4.6](#) gives a lower bound on (a term in) the denominator. Plugging those bounds in [\(15\)](#),

$$\sum_{j>r} \rho_j \leq \frac{2\varepsilon^{-1}\sqrt{d_v}}{\sqrt{r} \cdot \varepsilon^{3/2}/\sqrt{8}} \sum_{j \leq n} \rho_j \leq \frac{1}{\sqrt{r}} \cdot \frac{6\sqrt{d_v}}{\varepsilon^{5/2}} \cdot \sum_{j \leq n} \rho_j. \quad (16)$$

279 Suppose $r > 144\varepsilon^{-5}d_v$. Then $\sum_{j>r} \rho_j < (1/2) \sum_{j \leq n} \rho_j$. The sweep cut C is constructed with the smallest
 280 value of r such that $\sum_{j>r} \rho_j < (1/2) \sum_{j \leq n} \rho_j$. Hence, $|C| \leq 144\varepsilon^{-5}d_v$. $\square$

281 An additional technical claim we need bounds the triangle weight incident to a single vertex.

282 **Claim 4.9.** *For all vertices $u \in V(H)$, $\text{wt}(T_H(u)) \leq (2d_v)^{-1}$.*

Proof. Consider edge $(u, w) \in E(H)$. We will prove that $\text{wt}(T_H((u, w))) \leq d_u^{-1}d_v^{-1}$. Recall that d_v is the smallest degree among vertices in H . Furthermore, $|T_H((u, w))| \leq d_w$, since the third vertex in a triangle containing (u, w) is a neighbor of w .

$$\text{wt}(T_H((u, v))) = \sum_{z:(z,u,w) \in T(H)} \frac{1}{d_u d_w d_z} \leq \frac{1}{d_u d_v} \sum_{z:(z,u,w) \in T(H)} \frac{1}{d_w} \leq \frac{1}{d_u d_v} \times \frac{d_w}{d_w} = \frac{1}{d_u d_v}$$

We now bound $\text{wt}(T_H(u))$ by summing over all neighbors of u in H .

$$\begin{aligned} \text{wt}(T_H(u)) &= (1/2) \sum_{w:(u,w) \in E(H)} \text{wt}(T_H((u, w))) \\ &\leq (1/2) \sum_{w:(u,w) \in E(H)} \frac{1}{d_u d_v} = \frac{1}{2d_v} \sum_{w:(u,w) \in E(H)} \frac{1}{d_u} \\ &\leq \frac{1}{2d_v} \times \frac{d_u}{d_u} = \frac{1}{2d_v}. \end{aligned}$$

283 $\square$

284 4.1 The proof of [Theorem 4.2](#)

285 *Proof.* (of [Theorem 4.2](#)) By construction of X as $\{v\} \cup L \cup C$, all the triangles of the form (w, u, u') , where
 286 $w \in C$ and $u, u' \in L$, are contained in X . The total weight of such triangles is at least $\sum_{v \leq n} \rho_v/2$, by the
 287 construction of C . By [Claim 4.6](#), $\sum_{v \leq n} \rho_v/2 \geq \varepsilon^3/16$.

288 Let us now bound that total triangle weight incident to X in H . Observe that $|X| = 1 + |L| + |C|$ which
 289 is at most $1 + d_v + \varepsilon^{-5}144d_v$, by [Claim 4.8](#). We can further bound $|X| \leq \varepsilon^{-5}146d_v$. By [Claim 4.9](#), the total
 290 triangle weight incident to a vertex is at most $(2d_v)^{-1}$. Hence, the total triangle weight incident to all of X
 291 is at most $73\varepsilon^{-5}$.

292 Thus, the triangle weight contained in X is at least $\frac{\varepsilon^3/16}{73\varepsilon^{-5}}$ times the triangle weight incident to X . The
 293 ratio is at least $\varepsilon^8/2000$, completing the proof of the first statement.

Proof of uniformity of $\mathcal{A}|_X$: We first prove a lower bound on the uniformity of $\mathcal{A}|_X$. For convenience, let B denote the set $\{e \in E(H), e \subseteq L\}$. By [Claim 4.5](#), $\sum_{e \in B} \text{wt}(e) \geq \varepsilon^2/8$. There are at most $\binom{d_v}{2} \leq d_v^2/2$

edges in B . For every edge e , $\text{wt}(e) \leq 1/d_v^2$. Let k denote the number of edges in B whose weight is at least $\varepsilon^2/16$.

$$\begin{aligned} \frac{\varepsilon^2}{8} &\leq \sum_{\substack{e \in B \\ \text{wt}(e) \leq \varepsilon^2 d_v^{-2}/16}} \text{wt}(e) + \sum_{\substack{e \in B \\ \text{wt}(e) \geq \varepsilon^2 d_v^{-2}/16}} \text{wt}(e) \\ &\leq |B| \times \varepsilon^2 d_v^{-2}/16 + k d_v^{-2} \\ &\leq d_v^2 \times \varepsilon^2 d_v^{-2}/16 + k d_v^{-2} \\ &= \varepsilon^2/16 + k d_v^{-2}. \end{aligned}$$

294 Rearranging, $k \geq \varepsilon^2 d_v^2/16$.

Hence, there are at least $\varepsilon^2 d_v^2/16$ edges contained in X with weight at least $\varepsilon^2 d_v^{-2}/16$. Consider the random variable Z that is the weight of a uniform random edge contained in X . Since $|X| \leq \varepsilon^{-5} 144 d_v$, the number of edges in X is at most $\varepsilon^{-10} (144)^2 d_v^2$. So,

$$\mathbf{E}[Z] \geq \frac{\varepsilon^2 d_v^2/16}{\varepsilon^{-10} (144)^2 d_v^2} \times \varepsilon^2 d_v^{-2}/16 \geq 2\delta \varepsilon^{14} d_v^{-2}. \quad (17)$$

295 The maximum value of Z is the largest possible weight of an edge in $E(H)$, which is at most d_v^{-2} . Applying
296 the reverse Markov bound of [Lemma 3.8](#), $\Pr[Z \geq \delta \varepsilon^{14} d_v^{-2}] \geq \delta \varepsilon^{14}$. Thus, an ε^{14} fraction of edges in $|X|$ have
297 weight at least $\delta \varepsilon^{14} d_v^{-2} \geq \delta \varepsilon^c/|X|^2$. Moreover, every edge has weight at most $d_v^{-2} \leq 1/(\delta \varepsilon^c |X|^2)$. So we
298 prove the uniformity of $\mathcal{A}|_X$.

299 The largest possible weight for any edge in $E(H)$ is d_v^{-2} . The size of $|X|$ is at least d_v and at most
300 $\varepsilon^{-5} 144 d_v$. Hence, $\mathcal{A}|_X$ is at least $\delta \varepsilon^{12}$ -uniform.

301 **Proof of strong uniformity:** For strong uniformity, we need to repeat the above argument within
302 neighborhoods in X . We prove in the beginning of this proof that the total triangle weight inside X is at
303 least $\varepsilon^3/16$. We also proved that $|X| \leq 146 \varepsilon^{-5} d_v$. Consider the random variable Z that is the triangle weight
304 contained in X incident to a uniform random vertex in X . Note that $\mathbf{E}[Z] \geq (\varepsilon^3/16)/(146 \varepsilon^{-5} d_v) \geq 2\delta' \varepsilon^8 d_v^{-1}$.
305 By [Claim 4.9](#), Z is at most $(2d_v)^{-1}$. Applying [Lemma 3.8](#), $\Pr[Z \geq \delta' \varepsilon^8 d_v^{-1}] \geq \delta \varepsilon^8$. This means that at least
306 $\delta' \varepsilon^8 |X|$ vertices in X are incident to at least $\delta' \varepsilon^8 d_v^{-1}$ triangle weight inside X .

Consider any such vertex u . Let $N(u)$ be the neighborhood of u in X . Every edge e in $N(u)$ forms a triangle with u with weight $\text{wt}(e)/d_u$. Hence, noting that $d_u \geq d_v$,

$$\sum_{e \subseteq N(u)} \text{wt}(e) d_u^{-1} \geq \delta' \varepsilon^8 d_v^{-1} \implies \sum_{e \subseteq N(u)} \text{wt}(e) \geq \delta' \varepsilon^8. \quad (18)$$

307 There are at most $|X|^2 \leq \varepsilon^{-10} (146)^2 d_v^2$ edges in $N(u)$. Let Z denote the weight of a uniform random
308 edge in $N(u)$. Note that $\mathbf{E}[Z] \geq \delta' \varepsilon^8/(\varepsilon^{-10} (146)^2 d_v^2) \geq 2\delta \varepsilon^{18} d_v^{-2}$. The maximum weight of an edge is at
309 most d_v^{-2} . By [Lemma 3.8](#), at least $\delta \varepsilon^{18}$ fraction of edges in $N(u)$ have a weight of at least $\delta \varepsilon^{18} d_v^{-2}$. Since
310 $|N(u)| \leq |X| \leq \varepsilon^{-5} 146 d_v$, this implies that $N(u)$ is also $\delta \varepsilon^c$ -uniform. Hence, we prove strong uniformity as
311 well.

312 □

313 5 Obtaining the decomposition

314 We first describe the algorithm that obtains the decomposition promised in [Theorem 1.3](#).

315 We partition all the triangles of G into three sets depending on how they are affected by $\text{Decompose}(G)$.
316 (i) The set of triangles removed by the cleaning step of [Step 4](#), (ii) the set of triangles contained in some
317 $X_i \in \mathbf{X}$, or (iii) the remaining triangles. Abusing notation, we refer to these sets as T_C , T_X , and T_R
318 respectively. Note that the triangles of T_R are the triangles “cut” when X_i is removed.

319 **Claim 5.1.** $\text{wt}(T_C) \leq (\tau/6) \sum_{e \in E} \text{wt}(e)$.

Algorithm 2 Decompose(G)

```
1: Initialize  $\mathbf{X}$  to be an empty family of sets, and initialize subgraph  $H = G$ .
2: while  $H$  is non-empty do
3:   while  $H$  is not clean do
4:     Remove an edge  $e \in E(H)$  from  $H$  such that  $\text{wt}(T_H(e)) < (\varepsilon)\text{wt}(e)$ .
5:   end while
6:   Add output Extract( $H$ ) to  $\mathbf{X}$ .
7:   Remove these vertices from  $H$ .
8: end while
9: Output  $\mathbf{X}$ .
```

320 *Proof.* Consider an edge e removed at [Step 4](#) of **Decompose**. Recall that ε is set to $\tau/6$. At that removal, the
321 total weight of triangles removed (cleaned) is at most $(\tau/6)\text{wt}(e)$. An edge can be removed at most once, so
322 the total weight of triangles removed by cleaning is at most $(\tau/6) \sum_{e \in E} \text{wt}(e)$. $\square$

323 *Proof.* (of [Theorem 1.3](#)) Let us denote by $H_1, H_2, \dots, H_k$ the subgraphs of which **Extract** is called. Let the
324 output of **Extract**(H_i) be denoted X_i . By the uniformity guarantee of [Theorem 4.2](#), each $\mathcal{A}|_{X_i}$ is $\delta\tau^c$ -uniform.

It remains to prove the coverage guarantee. We now sum the bound of [Theorem 4.2](#) over all X_i . (For convenience, we expand out ε as $\tau/6$ and let δ' denote a sufficiently small constant.)

$$\sum_{i \leq k} \sum_{t \in T(H), t \subseteq X} \text{wt}(t) \geq (\delta'\tau^8) \sum_{i \leq k} \sum_{t \in T(H), t \cap X \neq \emptyset} \text{wt}(t). \quad (19)$$

The LHS is precisely $\text{wt}(T_X)$. Note that a triangle appears at most once in the double summation in the RHS. That is because if $t \cap X_i \neq \emptyset$, then t is removed when X_i is removed. Since H_i is always clean, the triangles of T_C cannot participate in this double summation. Hence, the RHS summation is $\text{wt}(T_X) + \text{wt}(T_R)$ and we deduce that

$$\text{wt}(T_X) \geq \delta'\tau^8(\text{wt}(T_X) + \text{wt}(T_R)) \quad (20)$$

Note that $\text{wt}(T_c) + \text{wt}(T_x) + \text{wt}(T_r) = \sum_{t \in T} \text{wt}(t)$. There is where the definition of τ makes its appearance. By [Lemma 3.5](#), we can write the above equality as $\text{wt}(T_c) + \text{wt}(T_x) + \text{wt}(T_r) = (\tau/3) \sum_{e \in E} \text{wt}(e)$. Applying [Claim 5.1](#), (20), and the relation of edge weights to the Frobenius norm ([Claim 3.2](#)),

$$(\delta'\tau^8)^{-1} \text{wt}(T_X) \geq (\tau/6) \sum_{e \in E} \text{wt}(e) \implies \text{wt}(T_X) \geq \delta\tau^c \|\mathcal{A}\|_2^2 \quad (\text{by } \text{Claim 3.2}) \quad (21)$$

325 By [Claim 3.4](#), $\sum_{i \leq k} \|\mathcal{A}|_{X_i}\|_2^2 \geq \text{wt}(T_X)$, completing the proof of the coverage bound. $\square$

326 6 Algorithmics and implementation

327 We discuss theoretical and practical implementations of the procedures computing the decomposition of
328 [Theorem 1.3](#). The main operation required is a triangle enumeration of G ; there is a rich history of algorithms
329 for this problem. The best known bound for sparse graph is the classic algorithm of Chiba-Nishizeki that
330 enumerates all triangles in $O(m\alpha)$ time, where α is the graph degeneracy.

331 We first provide a formal theorem providing a running time bound. We do not explicitly describe the
332 implementation through pseudocode, and instead explain the main details in the proof.

333 **Theorem 6.1.** *There is an implementation of **Decompose**(G) whose running time is $O(R + (m + n + T) \log n)$,
334 where R is the running time of listing all triangles. The space required is $O(T)$ (where T is the triangle count).*

335 *Proof.* We assume an adjacency list representation where each list is stored in a dictionary data structure
336 with logarithmic time operations (like a self-balancing binary tree).

We prepare the following data structure that maintains information about the current subgraph H . We initially set $H = G$. We will maintain all lists as hash tables so that elementary operations on them (insert, delete, find) can be done in $O(1)$ time.

1. A list of all triangles in $T(H)$ indexed by edges. Given an edge e , we can access a list of triangles in $T(H)$ containing e .
2. A list of $\text{wt}(T_H(e))$ values for all edges $e \in E(H)$.
3. A list U of all (unclean) edges such that $\text{wt}(T_H(e)) < \varepsilon \text{wt}(e)$.
4. A min priority queue Q storing all vertices in $V(H)$ keyed by degree d_v . We will assume pointers from v to the corresponding node in Q .

These data structures can be initialized by enumerating all triangles, indexing them, and preparing all the lists. This can be done in $O(R)$ time.

We describe the process to remove an edge from H . When edge e is removed, we go over all the triangles in $T(H)$ containing e . For each such triangle t and edge $e' \in t$, we remove t from the triangle list of e' . We then update $\text{wt}(T_H(e'))$ by reducing it by $\text{wt}(t)$. If $\text{wt}(T_H(e'))$ is less than $\text{wt}(e)$, we add it to U . Finally, if the removal of e removes a vertex v from $V(H)$, we remove v from the priority queue Q . Thus, we can maintain the data structures. The running time is $O(|T_H(e)|)$ plus an additional $\log n$ for potentially updating Q . The total running time for all edge deletes is $O(T + n \log n)$.

With this setup in place, we discuss how to implement **Decompose**. The cleaning operation in **Decompose** can be implemented by repeatedly deleting edges from the list U , until it is empty.

We now discuss how to implement **Extract**. We will maintain a max priority queue R maintaining the values $\{\rho_w\}$. Using Q as defined earlier, we can find the vertex v of minimum degree. By traversing its adjacency list in H , we can find the set L . We determine all edges in L by traversing the adjacency lists of all vertices in L . For each such edge e , we enumerate all triangles in H containing e . For each such triangle t and $w \in t$, we will update the value of ρ_w in R .

We now have the total $\sum_w \rho_w$ as well. We find the sweep cut by repeatedly deleting from the max priority queue R , until the sum of ρ_w values is at least half the total. Thus, we can compute the set X to be extracted. The running time is $O((|X| + |E(X)| + |T(X)|) \log n)$, where $E(X), T(X)$ are the set of edges and triangles incident to X .

Overall, the total time for all the extractions and resulting edge removals is $O((n + m + T) \log n)$. The initial triangle enumeration takes R time. We add to complete the proof. $\square$

Practical considerations: In our code implementation, we apply some simplifying heuristics. Instead of repeatedly cleaning using a list, we simply make multiples passes over the graph, deleting any edge that is unclean. On deletion of edge e , we do *not* update the $T_H(e)$ values. We only perform the update after a complete pass over the graph. We do two to three passes over the graph, and leave any unclean edges that still remain. Typically, the first two passes remove almost all unclean edges, and it is not worth the extra time to find all remaining unclean edges.

7 Empirical Validation

7.1 Datasets

We now present an empirical validation of [Theorem 1.3](#) and the procedure **Decompose**. We show that spectral triadic decompositions exist in real-world networks; moreover, the clusters of the decompositions are often semantically meaningful. We perform experiments on a number of real-world networks, whose details are listed in [Tab. 1](#). Most of our graphs are undirected, and the network names are indicative of what they are: names beginning with ‘ca’ refer to coauthorship networks (ca-CondMat is for researchers who work in condensed matter, ca-DBLP does the same for researchers whose work is on DBLP, a computer science bibliography website), ones beginning with ‘com’ are social networks (socfb-Rice31 is a Facebook network, soc-hamsterster is from Hamsterster, a pet social network), and ‘cit’ refers to citation networks. While citation networks are in reality directed graphs, we consider any directed edge to be an undirected edge for the

purposes of our experiments. Graphs have been taken from the SNAP dataset at <https://snap.stanford.edu/data/> [19] and the network repository at <https://networkrepository.com/> [25]. The exceptions to this are the cit-DBLP dataset, which has been taken from <https://www.aminer.org/citation> [32] and the ca-cond-matL dataset, which has been taken from [21]; the L is for labelled. Not all graphs are used for all tasks and datasets have been specified with the associated experiments; the majority of the quantitative evaluation has been done on the first four graphs. The ground truth results have been performed on the ca-DBLP graph, and the last two are used to exhibit semantic sensibility of the extracted clusters.

Implementation details: The code is written in Python, and we run it on Jupyter using Python 3.7.6 on a Dell notebook with an Intel i7-10750H processor and 32 GB of ram. The code requires enough storage to store all lists of triangles, edges and vertices, and may be found on github at <https://bitbucket.org/Boshu1729/triadic/src/master/>. We set the parameter ε to 0.1 for all the experiments, unless stated otherwise. In general, we observe that the results are stable with respect to this parameter, and it is convenient choice for all datasets.

The $\tau(G)$ values and spectral triadic decompositions: In Tab. 1, we list the spectral triadic content, $\tau(G)$, of the real-world networks. Observe that they are quite large. They are the highest in social networks, consistently ranging in values greater than 0.1. This shows the empirical significance of $\tau(G)$ in real-world networks, which is consistent with large clustering coefficients.

In Tab. 2 we list the minimum and 10th percentile uniformity of the clusters in the decomposition (if the uniformity is, say 0.1, it means that at least a 0.1-fraction of entries in the submatrix have value at least 0.1 times the average value). We discuss these results in later sections as well; but the high uniformity in extracted clusters is a good indicator of the efficacy of the algorithm.

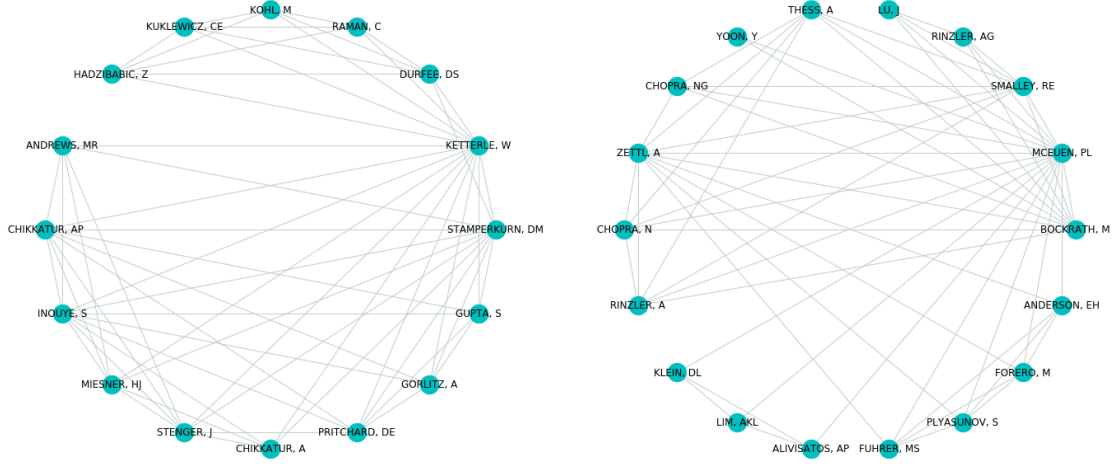
Relevance of decomposition: For the ca-DBLP graph, we do a detailed analysis of the clusters X_i of the decomposition with respect to a ground truth community labeling. We consider the first 10000 clusters extracted and investigate their quality. The ground truth here is defined by publication venues, and we restrict the evaluation to the top 5000 ground truth communities as described in [40], where the authors curate a list of 5000 communities that they found worked well with community detection algorithms.

For each set X_i , we find the ground truth community of highest Jaccard similarity with X_i . We plot the histogram of Jaccard similarities in Fig. 8. We ignore clusters that have fewer than 5 vertices for the purposes of this experiment; this accounts for only 42 of the total clusters extracted. For the remainder, we observe that the mean Jaccard density is 0.3, and 46 communities have a perfect value of 1. Moreover, we plot a similar histogram for size of intersection of our clusters with the ground truth. Here too we observe that the mean is 6.48. We look at how this average varies with sizes of the extracted clusters in Fig. 9. While there is no clear trend observed in Jaccard density by cluster size, the mean intersection size clearly grows as we look at larger clusters.

Details of clusters: A spectral triadic decomposition produces a large number of approximately uniform dense clusters, starting from only the promise of a large $\tau(G)$ value. In Fig. 11 and Fig. 12, we show a scatterplot of clusters, with axes of cluster size versus uniformity across various networks. We see that there are a large number of fairly large clusters (of size at least 20) and of uniformity at least 0.5; further discussion on the clusters and their sizes is in Tab. 3. These plots are further validation of the significance of Theorem 1.3 and the utility of the spectral triadic decomposition. The procedure `Decompose` automatically produces a large number of approximately uniform (or assortative) blocks in real-world networks. We summarize the data with some numbers in Tab. 2. We also plot the edge density and triangle density of these clusters, which are more standard parameters in network science. Refer to the first two rows of Fig. 10 respectively for these plots. Since edge density is at least the uniformity, as expected, we see a large number of dense clusters extracted by `Decompose`.

In Fig. 3, we show graph drawings of two example clusters in a co-authorship network of (over 90K) researchers in Condensed Matter Physics. The cluster on the left has 16 vertices and 58 edges, and has

extracted a group of researchers who specialize in optics, ultra fast atoms, and Bose-Einstein condensates. Notable among them is the 2001 physics Nobel laureate Wolfgang Ketterle. The cluster on the right has 18 vertices and 55 edges, and has a group of researchers who all work on nanomaterials; there are multiple prominent researchers in this cluster, including the 1996 chemistry Nobel laureate Richard Smalley, who discovered buckminsterfullerene. We stress that the our decomposition found more than a *thousand* such clusters. Similar extracted clusters of research papers and articles extracted from the DBLP citation network



(a) Condensed Matter Physics: Cluster of researchers working on optics, ultra fast atoms, and Bose-Einstein condensates (b) Condensed Matter Physics: Cluster of researchers working on graphene, nanomaterials and topological insulators

Figure 3: We show two example clusters from a spectral triadic decomposition of coauthorship network of researchers in Condensed Matter Physics [21], a graph with $\tau = 0.25$. The left cluster is a set of 16 researchers (58 edges) working on optics and Bose-Einstein condensates (notably, the cluster has the 2001 Physics Nobel laureate Wolfgang Ketterle). The right cluster has 18 researchers (55 edges) working on nanomaterials, including the 1996 Chemistry Nobel laureate Richard Smalley.

can be seen in Fig. 4. In this case, one cluster is a group of papers on error correcting/detecting codes, while the other is a cluster of logic program and recursive queries papers.

It is surprising how well the spectral triadic decomposition finds fine-grained structure in networks, based on just the spectral transitivity. This aspect highlights the practical relevance of spectral theorems that decompose graphs into many blocks, rather than the classic Cheeger-type theorems that one produce two blocks.

Total content of decomposition: Even though `Decompose` does not explicitly optimize for it, the clusters capture a large fraction of the vertices, and triangles. In Tab. 3, we see the latter values for all the decompositions constructed. A significant fraction of both vertices and the total triangle weight is preserved. Coverage is also impressive across the board; this is the total Frobenius norm of the decomposition, as a fraction of the total Frobenius norm of $\mathcal{A}$. The cluster sizes vary with the dataset; the Facebook network shows especially large clusters; it also exhibits lower triangle weight retention, which may be an artefact of the fact that it is easier to retain triangle density in smaller clusters. A distribution of cluster sizes across datasets is shown in the histograms in the bottom row of Fig. 10.

that participate in cell division. Similarly, we have another group of 33 proteins in the second; including tightly bound groups of cytochromes, ATPG and NDHs that participate in aerobic respiration.

The second group in Fig. 7 has two such clusters from *Streptococcus*. The first is a smaller cluster of 11 proteins that participate in the energy-coupling factor transport system. The second is a dense cluster of 37 proteins that participate in the phosphoenolpyruvate phosphotransferase system (PTS), the primary mechanism by which bacteria such as *Streptococcus* transport sugars; a large number of these are specifically associated with the cellobiose PTS.

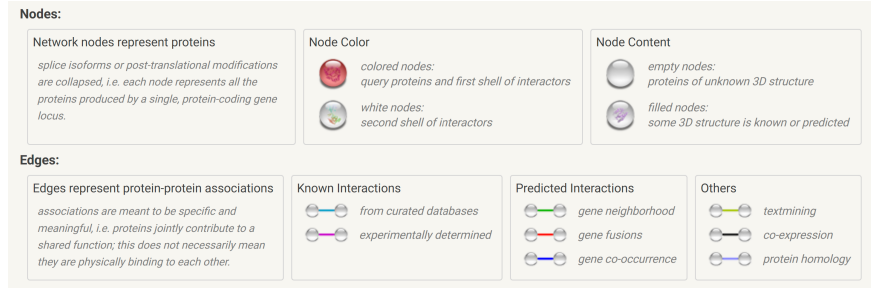


Figure 5: The key to the visuals for protein complexes, taken from STRINGdb. The indicators include color of node, color of edges, and content of nodes. Multiple edges in different colors denote different kinds of interaction; two proteins may have multiple simultaneous interactions.

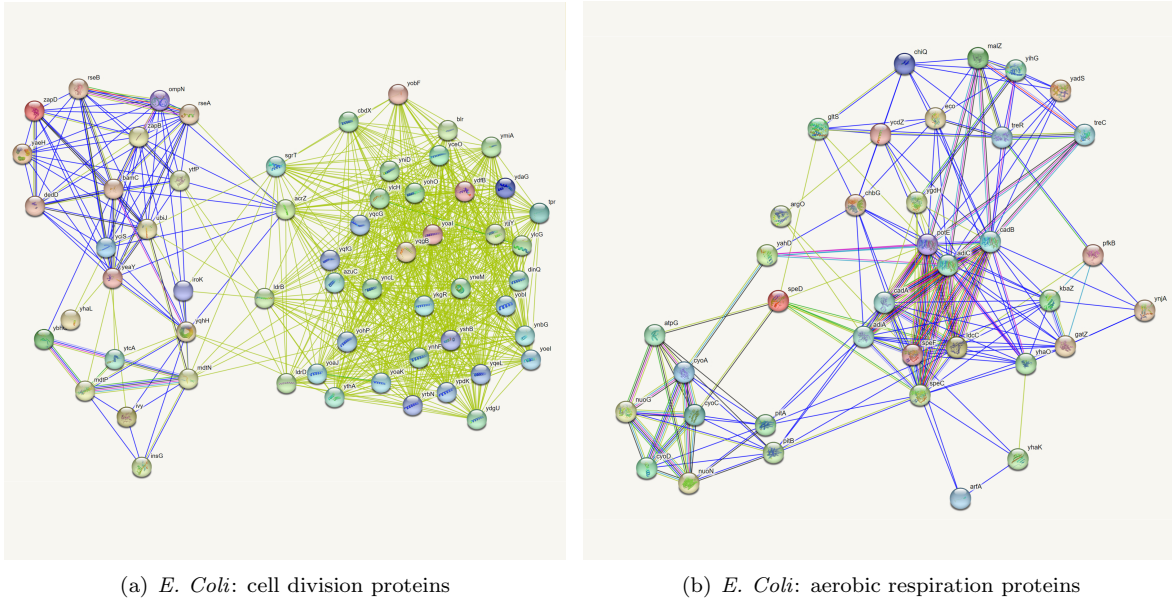


Figure 6: We show example clusters from a spectral triadic decomposition of protein-protein interaction network of the *E. Coli* K12 substrain MG 1655 [31]. The cluster on the left is a group of 60 proteins associated with cell division, and the one on the right is a cluster of 33 proteins associated with aerobic respiration.

Complement of CDF of size of intersection of clusters
with ground truth across discovered cluster size

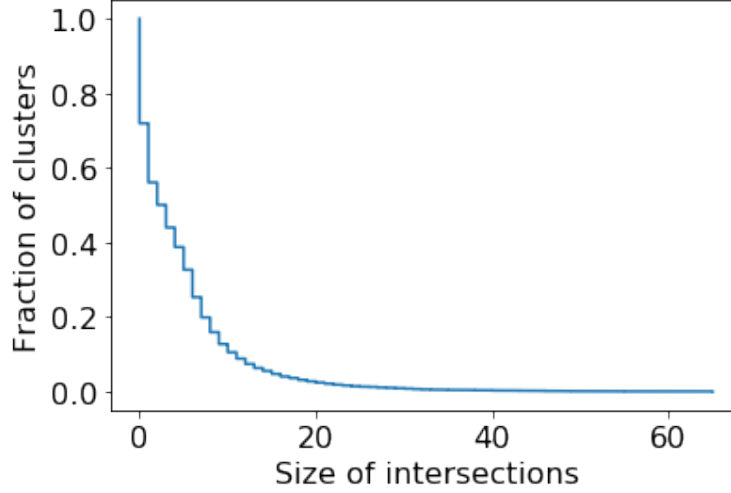


Figure 8: This figure looks at the quality of our extracted clusters compared with ground truth data. We compare results for the first 10000 communities extracted by our algorithm, and look at intersection size, excluding clusters with fewer than 5 vertices. We compare our clusters with the 5000 ‘high quality’ clusters from [40].

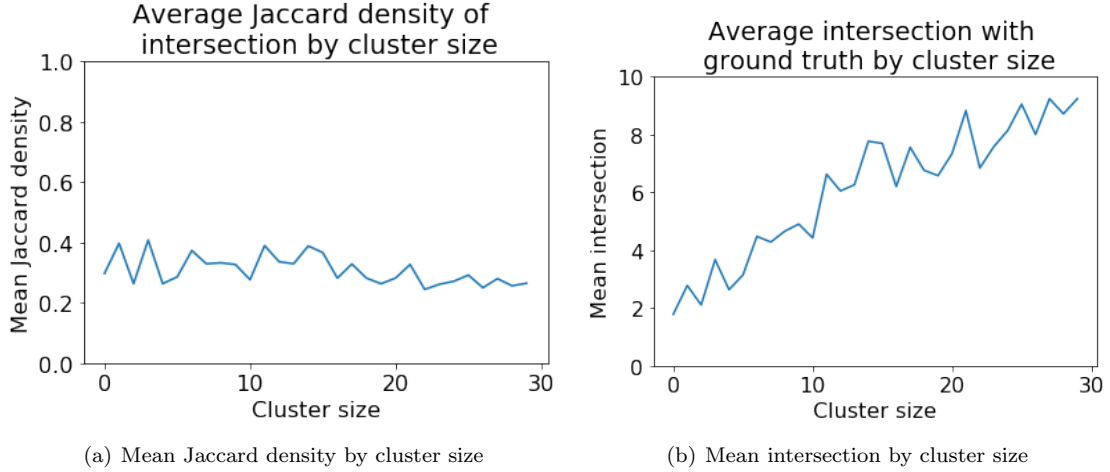


Figure 9: Here we look at the mean values of Jaccard density and intersection size across clusters of different sizes. We compare results for the first 10000 communities extracted by our algorithm, and look at Jaccard density and intersection size, excluding clusters with fewer than 5 vertices. We compare our clusters with the 5000 ‘high quality’ clusters from [40].

Dataset	Mean Uniformity	10th percentile	Min uniformity
soc-hamsterter	0.67	0.27	0.14
socfb-Rice31	0.24	0.15	0.08
ca-HepTh	0.28	0.22	0.11
ca-CondMat	0.28	0.21	0.08
ca-cond-matL	0.66	0.32	0.11
cit-HepTh	0.39	0.22	0.11
cit-DBLP	0.46	0.23	0.05

Table 2: Summary of data about the extracted clusters across datasets: number of clusters, percentage of total number of vertices preserved in clusters, total triangle weight preserved in clusters, minimum of cluster sizes, maximum of cluster sizes, average of cluster sizes when $\varepsilon = 0.1$.

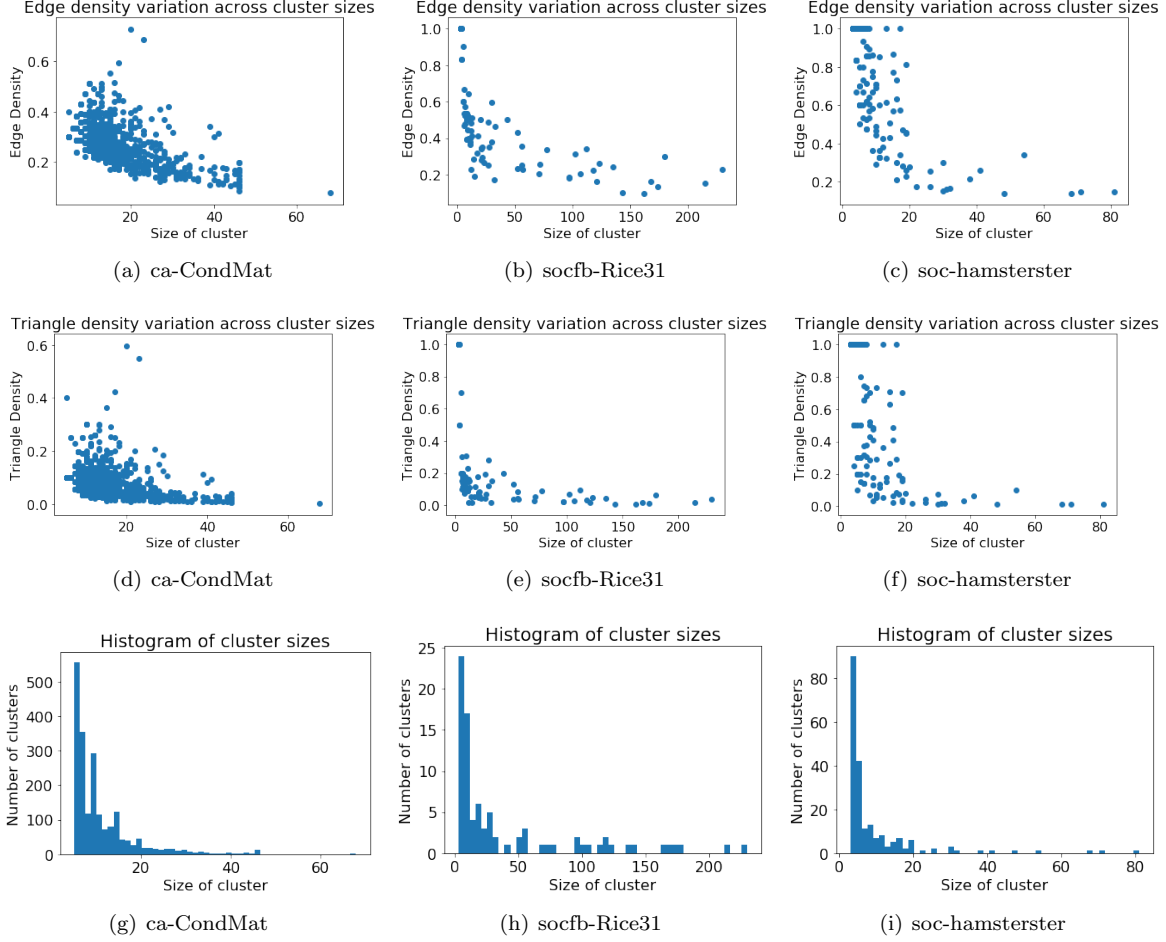


Figure 10: Scatter plot for edge density (top, (a)-(c)), triangle density (center, (d)-(f)) as a function of cluster size, and histogram of cluster sizes (bottom, (g)-(i)) for ca-CondMat, socfb-Rice31, and soc-hamsterster respectively, for $\varepsilon = 0.1$.

Dataset	#Clusters	% Vtx	% Tri-Wt	Coverage %	Cluster Sizes		
					Min	Max	Avg
soc-hamsterster	208	76.09	80.94	85.34	3	81	8.88
socfb-Rice31	86	86.84	24.71	36.76	3	230	41.27
ca-HepTh	849	77.46	71.43	73.79	5	47	9.01
ca-CondMat	2049	95.45	71.61	58.84	5	68	10.78
ca-condmatL	1566	75.57	77.90	78.64	3	47	7.85
cit-HepTh	1664	73.74	53.81	58.84	3	79	12.31
cit-DBLP	7265	27.57	70.04	77.15	3	111	8.25

Table 3: Summary of data about the extracted clusters across datasets: number of clusters, percentage of total number of vertices preserved in clusters, total triangle weight preserved in clusters, coverage, and cluster sizes (minimum, maximum and average). We observe great diversity in the nature of clusters extracted; most datasets have average cluster sizes between 7 and 13, with the exception of socfb-Rice31, where the average is as high as 41. Number of vertices preserved is consistently high except for the cit-DBLP network, which had remarkably low edge and triangle density to begin with. Triangle weight preserved and coverage are also remarkably high across all datasets; albeit a bit lower in socfb=Rice31.

Dataset	#Vertices	#Edges	#Triangles	τ
soc-hamsterster	2,427	16,630	53,251	0.215
socfb-Rice31	4,088	184,828	1,904,637	0.122
caHepTh	9,877	24,827	28,339	0.084
ca-cond-matL	16,264	47,594	68,040	0.255
ca-CondMat	23,133	93,497	176,063	0.125
cit-HepTh	27,770	352,807	1,480,565	0.122
cit-DBLP	217,312	632,542	248,004	0.087
ca-DBLP	317,080	1,049,866	2,224,385	0.248

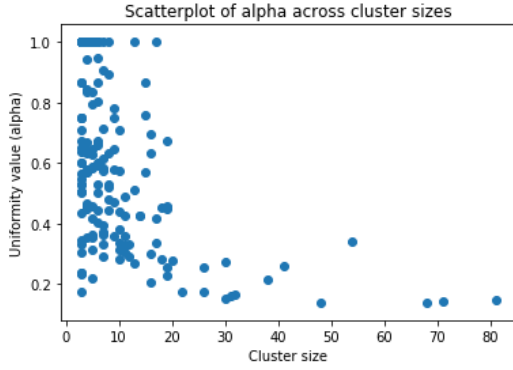
Table 1: Summary of datasets used for different experiments. Most of our graphs are undirected, and the network names are indicative of what they are: names beginning with ‘ca’ refer to coauthorship networks (ca-CondMat is for researchers who work in condensed matter, ca-DBLP does the same for researchers whose work is on DBLP, a computer science bibliography website), ones beginning with ‘com’ are social networks (socfb-Rice31 is a Facebook network, soc-hamsterster is from Hamsterster, a pet social network), and ‘cit’ refers to citation networks. While citation networks are in reality directed graphs, we consider any directed edge to be an undirected edge for the purposes of our experiments. Graphs have been taken from the SNAP dataset at <https://snap.stanford.edu/data/> [19] and the network repository at <https://networkrepository.com/> [25]. The exceptions to this are the cit-DBLP dataset, which has been taken from <https://www.aminer.org/citation> [32] and the ca-cond-matL dataset, which has been taken from [21] ; the L is for labelled. For cit-DBLP, the τ value is for the 2-core of the graph.

ε	#Clusters	% Vtx	% Tri-Wt	Cluster Min	Cluster Max	Cluster Avg
0.1	849	11.13	58.63	5	47	9.01
0.2	866	10.56	59.04	5	57	8.39
0.3	652	8.08	54.99	5	48	8.52
0.5	296	3.85	36.00	5	61	8.95

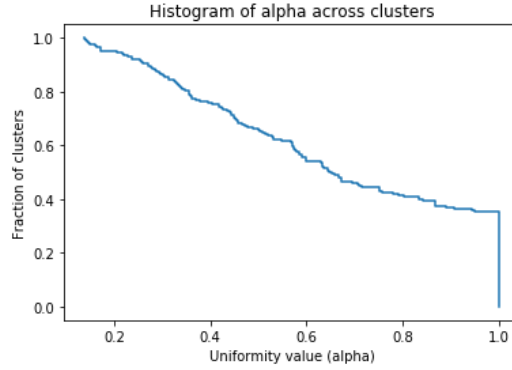
Table 4: Summary of data about the extracted clusters on caHepTh for $\varepsilon \in \{0.1, 0.2, 0.3, 0.5\}$: number of clusters, percentage of total number of vertices preserved in clusters, total triangle weight preserved in clusters, minimum of cluster sizes, maximum of cluster sizes, average of cluster sizes.

Paper Title	Venue	Year
Some comments on the aims of MIRFAC	Communications of the ACM	1964
MIRFAC: a compiler based on standard mathematical notation and plain English	Communications of the ACM	1963
MIRFAC: a reply to Professor Dijkstra	Communications of the ACM	1964
More on reducing truncation errors	Communications of the ACM	1964
The dangling else	Communications of the ACM	1964
MADCAP: a scientific compiler for a displayed formula textbook language	Communications of the ACM	1961
Further comment on the MIRFAC controversy	Communications of the ACM	1964
An experiment in a user-oriented computer system	Communications of the ACM	1964
Automatic programming and compilers II: The COLASL automatic encoding system	Proceedings of the 1962 ACM national conference on Digest of technical papers	1962

Table 5: Metadata for cluster extracted from cit-DBLP: Cluster of size 9, edge density of 0.472



(a) Scatter plot for uniformity

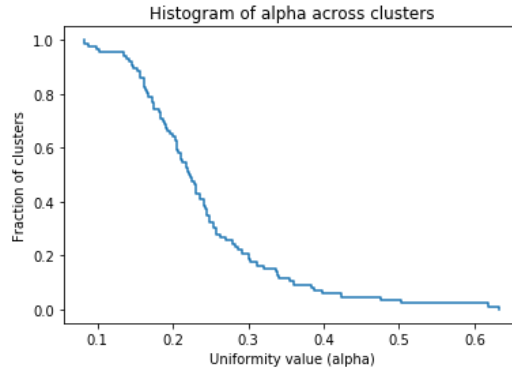


(b) Histogram for uniformity

soc-hamsterster

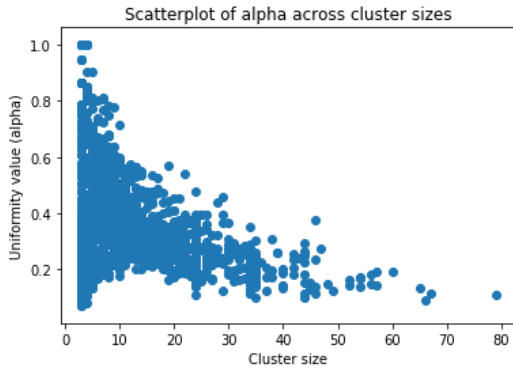


(c) Scatter plot for uniformity

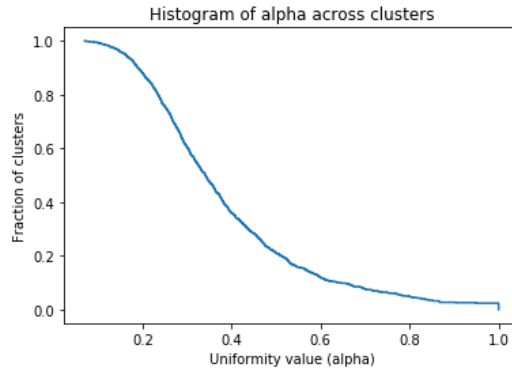


(d) Histogram for uniformity

socfb-Rice31



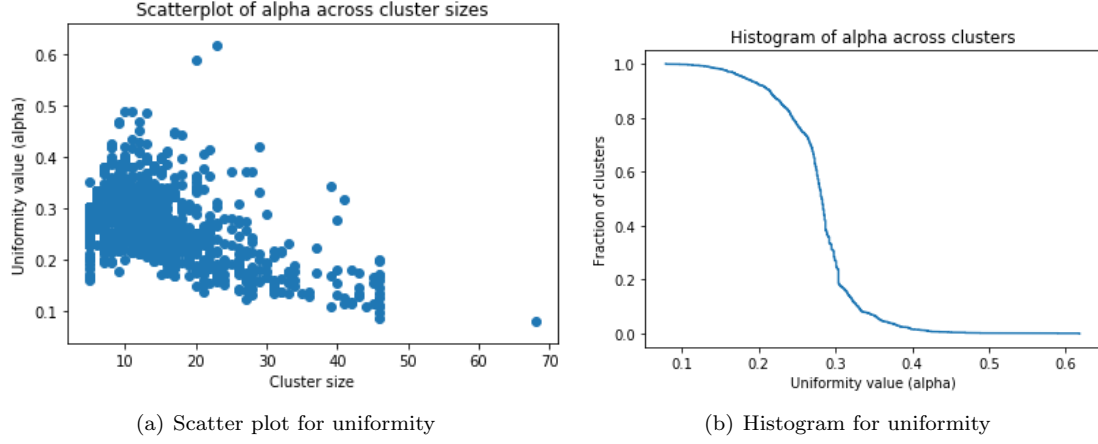
(e) Scatter plot for uniformity



(f) Histogram for uniformity

cit-HepTh

Figure 11: A look at uniformity across clusters in the decomposition obtained from various networks as labelled. The figures on the left are straightforward scatter plots that looks at uniformity across clusters of varying sizes. Those on the right are complementary cumulative histograms for the uniformity values. The x -axis is the uniformity value, and the y axis the fraction of clusters with *at least* that uniformity value.



ca-CondMat



ca-cond-matL



cit-DBLP

Figure 12: (Continued) A look at uniformity across clusters in the decomposition obtained from various networks as labelled. The figures on the left are straightforward scatter plots that look at uniformity across clusters of varying sizes. Those on the right are complementary cumulative histograms for the uniformity values. The x -axis is the uniformity value, and the y axis the fraction of clusters with *at least* that uniformity value.

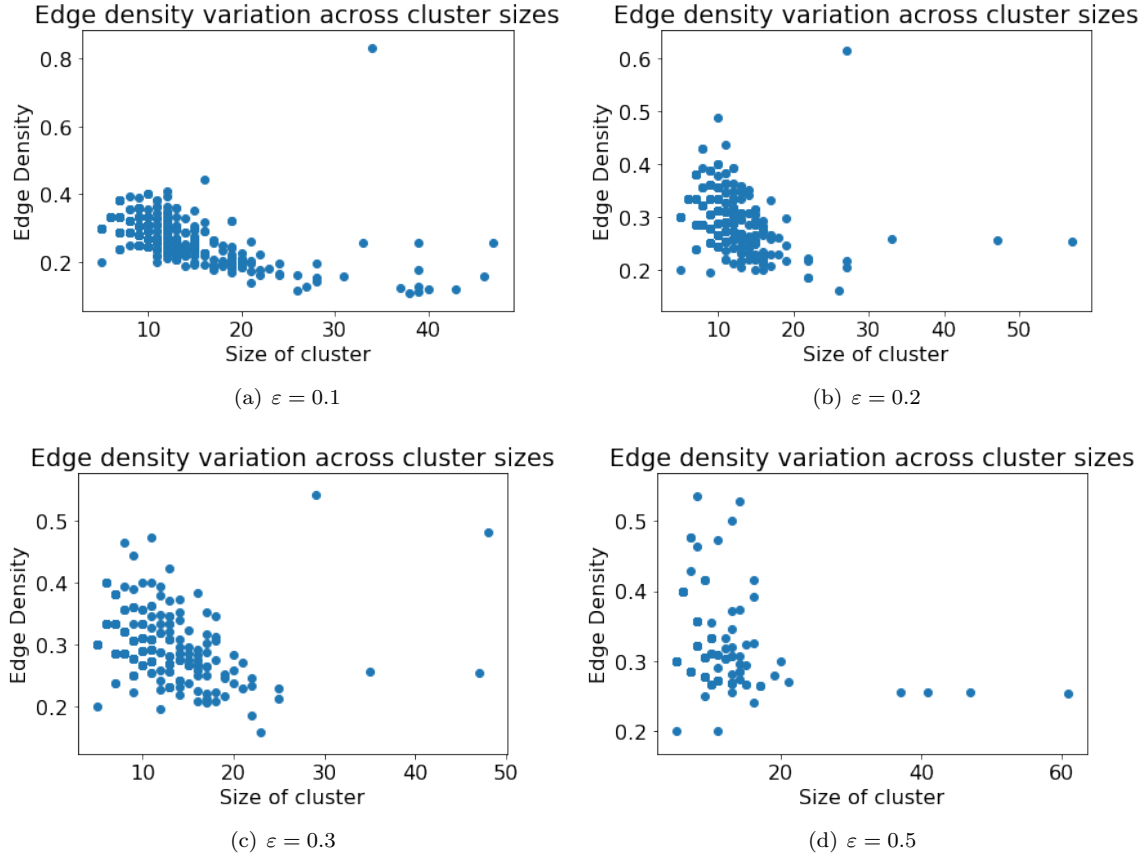


Figure 13: Scatter plot for edge density for ca-CondMat with varying values of ε .

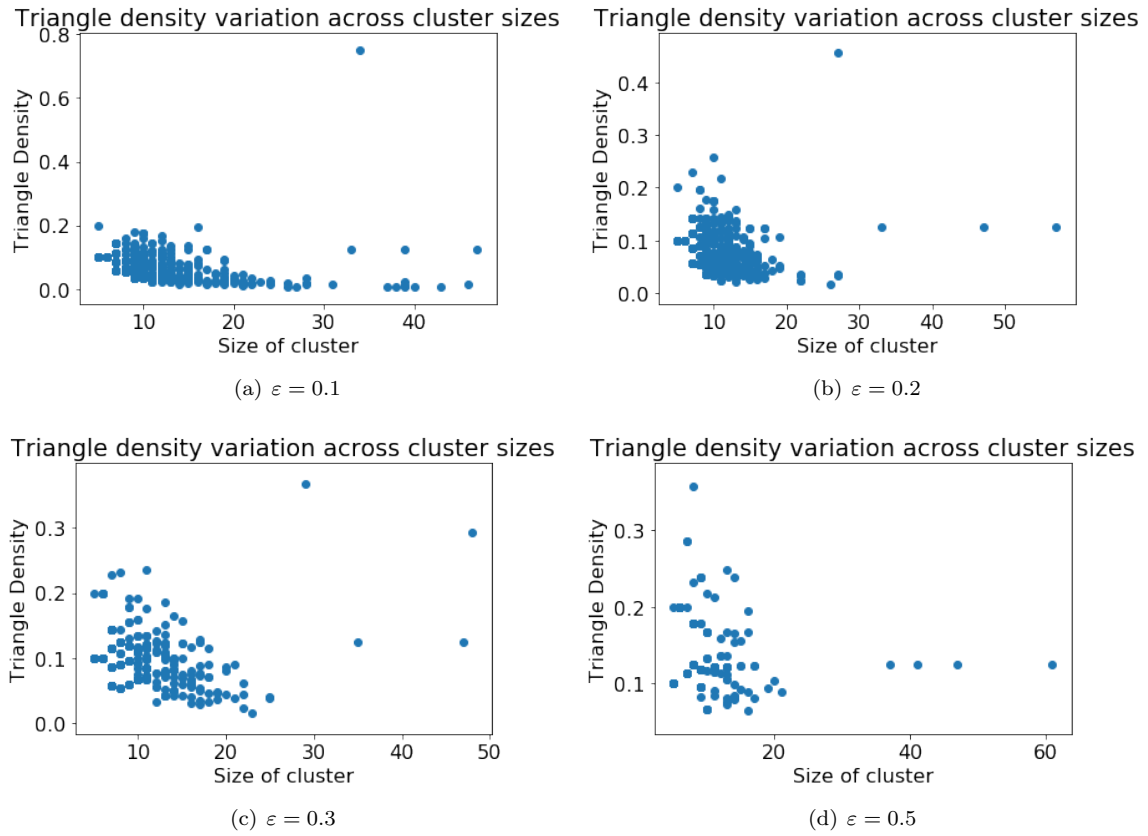


Figure 14: Scatter plot for triangle density for ca-CondMat with varying values of ε .

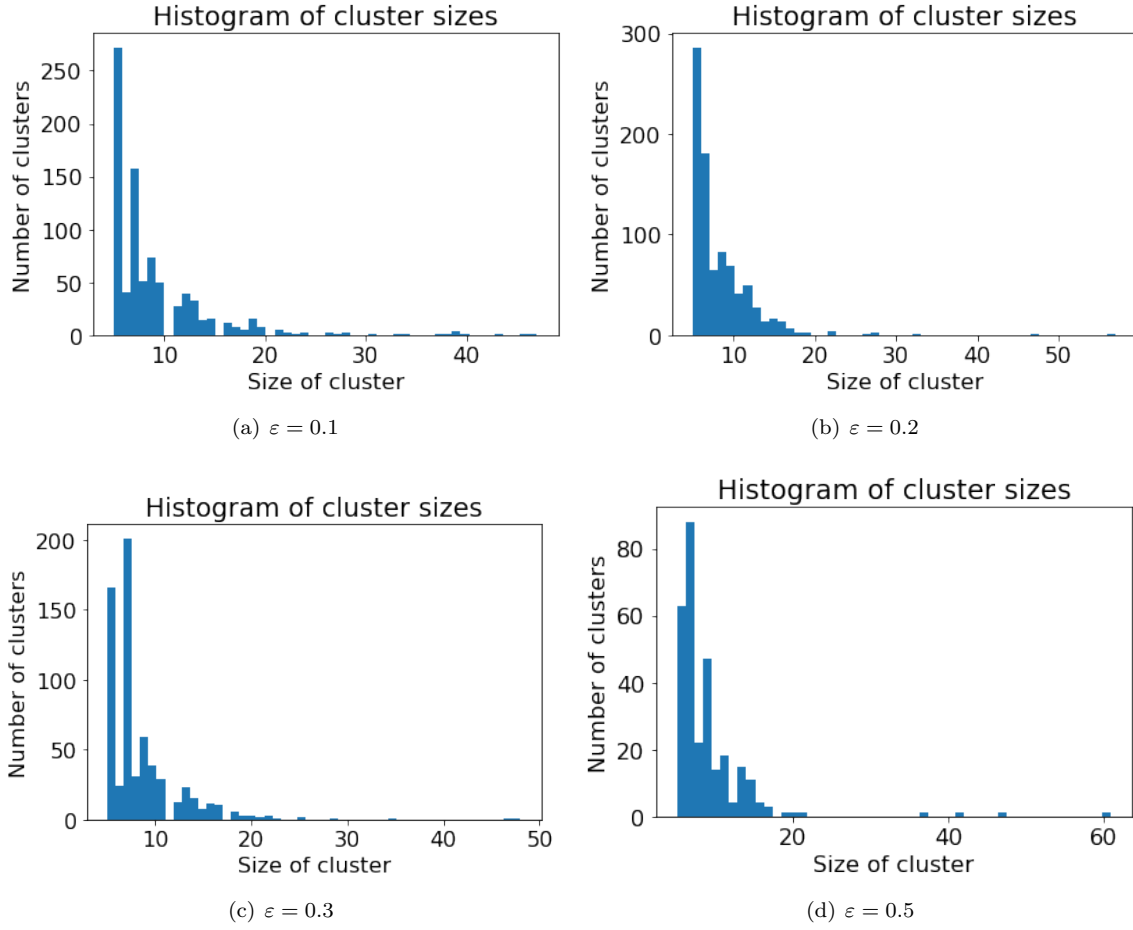


Figure 15: Histogram of cluster sizes for ca-CondMat with varying values of ε .

Paper Title	Venue	Year
Measurement-based characterization of IP VPNs	IEEE/ACM Transactions on Net- working (TON)	2007
Traffic matrices: balancing measurements, inference and modeling	Proceedings of the 2005 ACM SIG- METRICS international conference on Measurement and modeling of computer systems	2005
Data streaming algorithms for accurate and efficient measurement of traffic and flow matrices	ACM SIGMETRICS Performance Evaluation Review	2005
An information-theoretic approach to traffic matrix estimation	Proceedings of the 2003 conference on Applications, technologies, archi- tectures, and protocols for computer communications	2003
Atomic Decomposition by Basis Pursuit	SIAM Review	2001
Solving Ill-Conditioned and Singular Linear Systems: A Tutorial on Regularization	SIAM review	1998
Structural analysis of network traffic flows	ACM Sigmetrics performance evalu- ation review	2004
How to identify and estimate the largest traffic matrix elements in a dynamic environment	Proceedings of the joint interna- tional conference on Measurement and modeling of computer systems	2004
Relative information: theories and applications	<i>Book</i>	1990
Estimating point-to-point and point-to-multipoint traffic matri- ces: an information-theoretic approach	IEEE/ACM Transactions on Net- working (TON)	2005
Traffic matrix tracking using Kalman filters	ACM Sigmetrics performance evalu- ation review	2005
Towards a meaningful MRA of traffic matrices	Proceedings of the 8th ACM SIG- COMM conference on Internet mea- surement	2008

Table 6: Metadata for cluster extracted from cit-DBLP: Cluster of size 12, edge density of 0.83

Paper Title	Venue	Year
A Cell ID Assignment Scheme and Its Applications	Proceedings of the 2000 International Workshop on Parallel Processing	2000
High-Performance Computing on a Honeycomb Architecture	Proceedings of the Second International ACPC Conference on Parallel Computation	1993
Optimal dynamic mobility management for PCS networks	IEEE/ACM Transactions on Networking (TON)	2000
Higher dimensional hexagonal networks	Journal of Parallel and Distributed Computing	2003
Addressing and Routing in Hexagonal Networks with Applications for Tracking Mobile Users and Connection Rerouting in Cellular Networks	IEEE Transactions on Parallel and Distributed Computing	2002
Addressing, Routing, and Broadcasting in Hexagonal Mesh Multiprocessors	IEEE Transactions on Computers	1990
Performance Analysis of Virtual Cut-Through Switching in HARTS: A Hexagonal Mesh Multicomputer	IEEE Transactions on Computers	1990
HARTS: A Distributed Real-Time Architecture	Computer	1991
Cell identification codes for tracking mobile users	Wireless Networks	2002

Table 7: Metadata for cluster extracted from cit-DBLP: Cluster of size 9, edge density of 0.33

Paper Title	Venue	Year
Classical linear logic of implications	Mathematical Structures in Computer Science	2005
Logic continuations	Journal of Logic Programming	1987
Axioms for control operators in the CPS hierarchy	Higher-Order and Symbolic Computation	2007
Formalizing Implementation Strategies for First-Class Continuations	Proceedings of the 9th European Symposium on Programming Languages and Systems	2000
Linearly Used Effects: Monadic and CPS Transformations into the Linear Lambda Calculus	Proceedings of the 6th International Symposium on Functional and Logic Programming	2002
On Exceptions Versus Continuations in the Presence of State	Proceedings of the 9th European Symposium on Programming Languages and Systems	2000
What is a Categorical Model of Intuitionistic Linear Logic?	Proceedings of the Second International Conference on Typed Lambda Calculi and Applications	1995
Using a Continuation Twice and Its Implications for the Expressive Power of call/cc	Higher-Order and Symbolic Computation	1999
From control effects to typed continuation passing	Proceedings of the 30th ACM SIGPLAN-SIGACT symposium on Principles of programming languages	2003
Continuations: A Mathematical Semantics for Handling FullJumps	Higher-Order and Symbolic Computation	2000
Comparing Control Constructs by Double-Barrelled CPS	Higher-Order and Symbolic Computation	2002
Linear Continuation-Passing	Higher-Order and Symbolic Computation	2002
Definitional Interpreters for Higher-Order Programming Languages	Higher-Order and Symbolic Computation	1998
Essentials of programming languages	<i>Book</i>	1992
Glueing and orthogonality for models of linear logic	Theoretical Computer Science	2003
Frame rules from answer types for code pointers	Conference record of the 33rd ACM SIGPLAN-SIGACT symposium on Principles of programming languages	2006

Table 8: Metadata for cluster extracted from cit-DBLP: Cluster of size 16, edge density of 0.42

8 Comparisons with other methods

8.1 Shortcomings of Spectral k -way cut

In this section, we first look at some comparisons with the traditional spectral k -way clustering algorithm. The implementation here is the `scikit-learn` version, which uses the celebrated algorithm due to Ng, Jordan and Weiss [23]. The primary drawback of this procedure is that this gives no strong guarantees on internal density. Moreover, they are quite susceptible to minor changes in the graph: small perturbations can vastly alter the clusters obtained. We observe the results of the k -way clustering algorithm on some of our graphs: consistently, we notice that the algorithm overwhelmingly prefers a large single component, even for k as large as 100. The majority of the clusters obtained are otherwise quite small, often of size less than 3 or even singletons and disconnected. Moreover, even the edge densities in these components is remarkably low. Note that at such low values of edge density, triangle density is only further lower. This provides more evidence that while k -way clustering succeeds in some aspects, it does not perform particularly well if our objective is to find dense communities in networks.

Dataset	#Vertices	k	Clusters < 3	Clusters > 10	Max. Cluster	%Vertices in max	ED of max
soc-hamsterster	2,427	20	25.00%	40.00%	2,224	91.64%	0.0005
		60	8.33%	21.67%	1,960	80.67%	0.0006
		100	23.00%	10.00%	1,887	77.75%	0.0007
socfb-Rice31	4,088	20	25.00%	65.00%	1,745	39.97%	0.0076
		60	45.00%	33.33%	1,891	46.26%	0.0011
		100	54.00%	22.00%	1,598	39.09%	0.0016
ca-cond-matL	16,264	20	0.00%	40.00%	14,854	91.33%	7.372×10^{-5}
		60	6.67%	53.33%	14,535	89.37%	7.699×10^{-5}
		100	2.00%	46.00%	14,261	87.68%	7.998×10^{-5}
ca-CondMat	23,133	20	0.00%	65.00%	22,467	97.12%	4.583×10^{-5}
		60	5.00%	35.00%	22,132	95.67%	4.779×10^{-5}
		100	9.00%	19.00%	21,924	94.77%	4.813×10^{-5}

Table 9: Summary of data about the extracted clusters across datasets, ED refers to edge density and size refers to the number of vertices. The columns ‘Clusters < 3’ and ‘Clusters > 10’ respectively refer to the percentage of the (k) clusters that have size less than 3 and greater than 10, ‘Max. Cluster’ refers to the size of the largest cluster, ‘%Vertices in max’ is this size as a fraction of the entire graph, and ‘ED of max’ refers to the density in the largest cluster. We look at the variation in some metrics as we increase k . We observe that the algorithm, in its search for sparse cuts, ends up finding less meaningful clusters. Across all datasets, the majority (in some cases nearly all) the vertices have gone to a single large cluster, and most of the rest is in small clusters, often fewer than 3 vertices.

8.2 Shortcomings of the Louvain algorithm

The classic Louvain algorithm [4, 1] produces clusters with low uniformity value. We show scatter plots of the uniformity values in Fig. 16. Observe that, compared to the high uniformity values of the spectral triadic clusters, Louvain clusters have lower values. Hence, Louvain clusters are much less assortative. This is consistent with the literature that shows that Louvain’s clusters are often disconnected [33].

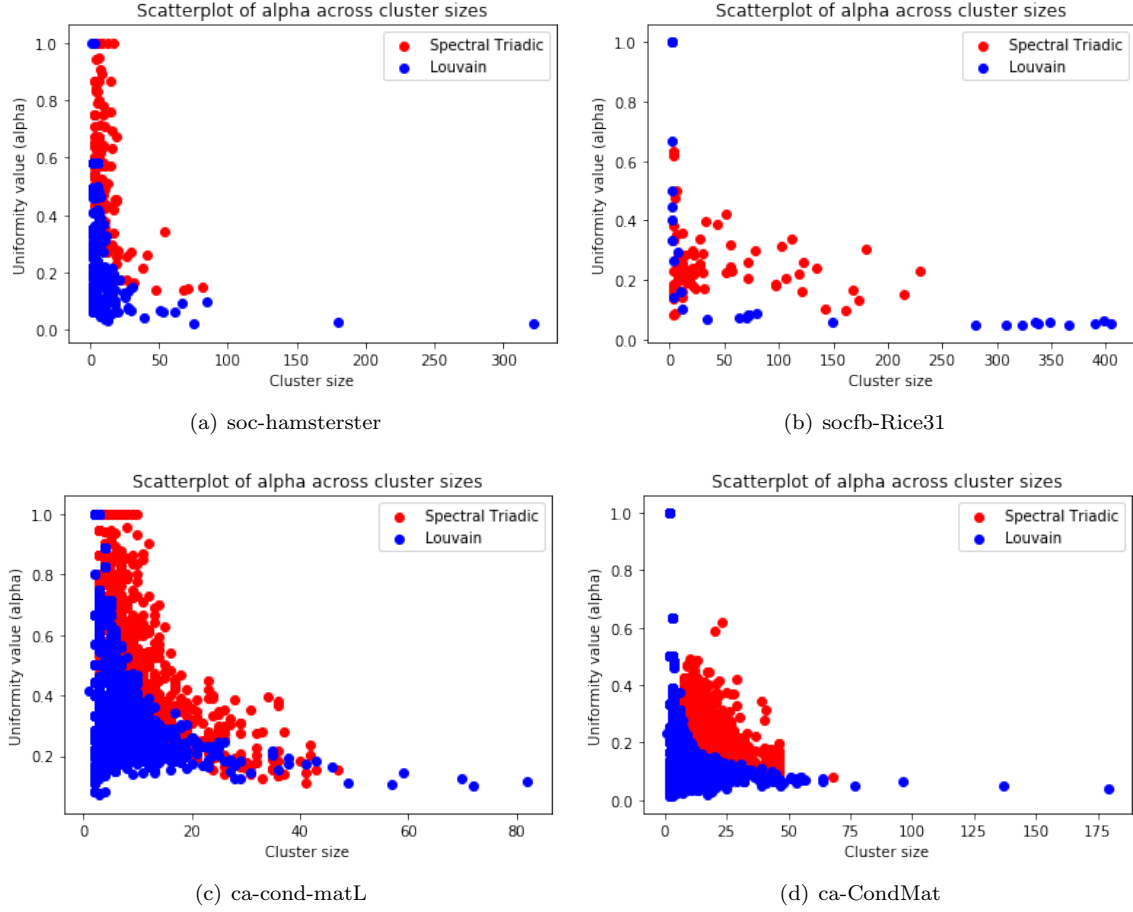


Figure 16: Scatter plots for uniformity (α) across different datasets, for both spectral triadic decompositions and the classic Louvain algorithm. We observe that the uniformity values of the Louvain clusters is significantly lower.

References

- [1] C++ code for louvain algorithm. <https://sourceforge.net/projects/louvain/>. 32
- [2] Reid Andersen, Fan Chung, and Kevin Lang. Local graph partitioning using pagerank vectors. pages 475–486, 2006. 6
- [3] A. Benson, D. F. Gleich, and J. Leskovec. Higher-order organization of complex networks. *Science*, 353(6295):163–166, 2016. 2, 4, 6
- [4] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, oct 2008. 32
- [5] Ronald S. Burt. Structural holes and good ideas. *American Journal of Sociology*, 110(2):349–399, 2004. 6
- [6] Deepayan Chakrabarti and Christos Faloutsos. Graph mining: Laws, generators, and algorithms. *ACM Comput. Surv.*, 38(1):2es, jun 2006. 5
- [7] F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997. 5
- [8] Katherine Faust. Comparing social networks: Size, density, and local structure. *Metodoloski zvezki*, 3:185–216, 07 2006. 2, 6
- [9] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3):75–174, 2010. 5
- [10] M. Girvan and M. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002. 5
- [11] D. Gleich and C. Seshadhri. Vertex neighborhoods, low conductance cuts, and good seeds for local community methods. pages 597–605, 2012. 6
- [12] M. Granovetter. The strength of weak ties: A network theory revisited. *Sociological Theory*, 1:201–233, 1983. 4
- [13] Rishi Gupta, Tim Roughgarden, and C. Seshadhri. Decompositions of triangle-dense graphs. *Innovations in Theoretical Computer Science*, pages 471–482, 2014. 5, 6
- [14] P. Holland and S. Leinhardt. A method for detecting structure in sociometric data. *American Journal of Sociology*, 76:492–513, 1970. 6
- [15] Paul W. Holland and Samuel Leinhardt. Local structure in social networks. *Sociological Methodology*, 7:1–45, 1976. 6
- [16] J. Kleinberg. Navigation in a small world. *Nature*, 406(6798), 2000. 2, 4
- [17] James R. Lee, Shayan Oveis Gharan, and Luca Trevisan. Multiway spectral partitioning and higher-order cheeger inequalities. *J. ACM*, 61(6):1–30, 2014. 2, 6
- [18] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics*, 6(1):29–123, 2009. 2, 5, 6
- [19] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. <http://snap.stanford.edu/data>, June 2014. 15, 22
- [20] S. Milgram. The small world problem. *Psychology Today*, 1(1):60–67, 1967. 4

- [21] M. E. J. Newman. The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences*, 98(2):404–409, 2001. 3, 15, 16, 22
- [22] M. E. J. Newman. Properties of highly clustered networks. *Phys. Rev. E*, 68:026121, Aug 2003. 5
- [23] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, NIPS’01, page 849856, Cambridge, MA, USA, 2001. MIT Press. 6, 32
- [24] Richard Peng, He Sun, and Luca Zanetti. Partitioning well-clustered graphs: Spectral clustering works! In *Conference on Learning Theory (COLT)*, volume 40 of *Proceedings of Machine Learning Research*, pages 1423–1455, 2015. 6
- [25] Ryan A. Rossi and Nesreen K. Ahmed. The network data repository with interactive graph analytics and visualization. *AAAI*, 2015. 3, 15, 22
- [26] A. Erdem Sariyuce, C. Seshadhri, A. Pinar, and U. Catalyurek. Finding the hierarchy of dense subgraphs using nucleus decompositions. In *World Wide Web (WWW)*, pages 927–937, 2015. 2, 6
- [27] C. Seshadhri, Tamara G. Kolda, and Ali Pinar. Community structure and scale-free collections of Erdos-Renyi graphs. *Physical Review E*, 85:056109, 2012. 2, 4, 5
- [28] Daniel A. Spielman. *Spectral and Algebraic Graph Theory*. <http://cs-www.cs.yale.edu/homes/spielman/sagt/>. 5
- [29] Daniel A. Spielman. Spectral graph theory and its applications. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 29–38, 2007. 5
- [30] Daniel A. Spielman and Shang-Hua Teng. A local clustering algorithm for massive graphs and its application to nearly linear time graph partitioning. *SIAM Journal on Computing*, 42(1):1–26, 2013. 6
- [31] D. Szklarczyk, A. Franceschini, S. Wyder, K. Forslund, D. Heller, J. Huerta-Cepas, M. Simonovic, A. Roth, A. Santos, K. P. Tsafou, M. Kuhn, P. Bork, L. J. Jensen, and C. von Mering. STRING v10: protein-protein interaction networks, integrated over the tree of life. *Nucleic Acids Res*, 43(Database issue):D447–452, Jan 2015. 17, 18, 19
- [32] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. Arnetminer: Extraction and mining of academic social networks. *SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, page 990998, 2008. 15, 19, 22
- [33] V.A. Traag, L. Waltman, and N.J. van Eck. From louvain to leiden: guaranteeing well-connected communities. *Scientific Reports*, 9(5233), 2019. 32
- [34] Amanda L Traud, Eric D Kelsic, Peter J Mucha, and Mason A Porter. Comparing community structure to characteristics in online collegiate social networks. *SIAM Rev.*, 53(3):526–543, 2011. 3
- [35] Amanda L Traud, Peter J Mucha, and Mason A Porter. Social structure of Facebook networks. *Phys. A*, 391(16):4165–4180, Aug 2012. 3
- [36] Charalampos E. Tsourakakis. The k-clique densest subgraph problem. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1122–1132, 2015. 2, 6
- [37] Charalampos E. Tsourakakis, Jakub Pachocki, and Michael Mitzenmacher. Scalable motif-aware graph clustering. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1451–1460, 2017. 2, 4, 6
- [38] Stanley Wasserman and Katherine Faust. *Social Network Analysis: Methods and Applications*. Structural Analysis in the Social Sciences. Cambridge University Press, 1994. 2

- 586 [39] D. Watts and S. Strogatz. Collective dynamics of ‘small-world’ networks. *Nature*, 393:440–442, 1998.
587 [2](#), [4](#), [6](#)
- 588 [40] Jaewon Yang and Jure Leskovec. Defining and evaluating network communities based on ground-truth.
589 *CoRR*, abs/1205.6233, 2012. [15](#), [17](#), [20](#)