Graph Summarization with Quality Guarantees

Matteo Riondato
Stanford University
rionda@cs.stanford.edu

David García-Soriano
Yahoo Labs, Barcelona, Spain
davidgs@yahoo-inc.com

Francesco Bonchi
Yahoo Labs, Barcelona, Spain
bonchi@yahoo-inc.com

***Abstract*—We study the problem of graph summarization. Given a large graph we aim at producing a concise lossy representation that can be stored in main memory and used to approximately answer queries about the original graph much faster than by using the exact representation. In this paper we study a very natural type of summary: the original set of vertices is partitioned into a small number of supernodes connected by superedges to form a complete weighted graph. The superedge weights are the edge densities between vertices in the corresponding supernodes. The goal is to produce a summary that minimizes the *reconstruction error* w.r.t. the original graph. By exposing a connection between graph summarization and geometric clustering problems (i.e., k-means and k-median), we develop the *first polynomial-time approximation algorithm* to compute the best possible summary of a given size.**

I. Introduction

Data analysts in several application domains (e.g., social networks, molecular biology, communication networks, and many others) routinely face graphs with millions of vertices and billions of edges. In principle, this abundance of data should allow for a more accurate analysis of the phenomena under study. However, as the graphs under analysis grow, mining and visualizing them become computationally challenging tasks. In fact, the running time of most graph algorithms depends on the size of the input: executing them on huge graphs might be impractical, especially when the input is too large to fit in main memory.

Graph summarization speeds up the analysis by creating a *lossy concise representation of the graph* that fits into main memory. Answers to otherwise expensive queries can then be computed using the summary without accessing the exact representation on disk. Query answers computed on the summary incur in a minimal loss of accuracy. Summaries can also be used for privacy purposes [1], to create easily interpretable visualizations of the graph [2], and to store a compressed representation of the graph.

LeFevre and Terzi [1] propose an enriched "supergraph" as a summary, associating an integer to each supernode and to each superedge, representing the number of edges (in the original graph) between vertices in the supernode and between the two sets of vertices connected by the superedge, respectively. From this lossy representation one can infer an *expected adjacency matrix*, where the expectation is taken over the set of *possible worlds* (i.e., graphs that are compatible with the summary). Thus, from the summary one can derive an approximated answer to a graph properties query as the expectation of the answer over the set of possible worlds.

The GraSS algorithm presented in [1] follows a greedy heuristic resembling an agglomerative hierarchical clustering using Ward's method [3] and as such can not give any guarantee on the quality of the summary. In this paper instead, we propose efficient algorithms to compute summaries of *guaranteed quality* (a constant factor from the optimal). This theoretical property is also verified empirically: our algorithms build more representative summaries and are much more efficient and scalable than GraSS in building those summaries.

II. Problem definition

We consider an undirected graph $G = (V, E)$ with $|V| = n$. In the rest of the paper, the key concepts are defined from the standpoint of the symmetric adjacency matrix A_G of G. We allow the edges to be weighted (so the adjacency matrix is not necessarily binary) and we allow self-loops (so the diagonal of the adjacency matrix is not necessarily all-zero).

Given a graph $G = (V, E)$ and $k \in \mathbb{N}$, a k-summary $\mathcal{S}$ of G is a *complete undirected weighted* graph $\mathcal{S} = (V', V' \times V')$ that is uniquely identified by a k-partition V' of V (i.e., $V' = \{V_1, \ldots, V_k\}$, s.t. $\cup_{i\in[1,k]} V_i = V$ and $\forall i, j \in [1, k], i \neq j$, it holds $V_i \cap V_j = \emptyset$). The vertices of $\mathcal{S}$ are called *supernodes*. There is a superedge e_{ij} for each unordered pair of supernodes (V_i, V_j), including (V_i, V_i) (i.e., each supernode has a self-loop e_{ii}). Each superedge e_{ij} is weighted by the density of edges between V_i and V_j:

$$d_G(i, j) = d_G(V_i, V_j) = e_G(V_i, V_j)/(|V_i||V_j|),$$

where for any two sets of vertices $S, T \subseteq V$, we denote

$$e_G(S, T) = \sum_{i\in S, j\in T} A_G(i, j).$$

We define the *density matrix* of $\mathcal{S}$ as the $k \times k$ matrix $A_{\mathcal{S}}$ with entries $A_{\mathcal{S}}(i, j) = d_G(i, j)$, $1 \leq i, j \leq k$. For each $v \in V$, we also denote by $s(v)$ the unique element of $\mathcal{S}$ (a supernode) such that $v \in s(v)$. The density matrix $A_{\mathcal{S}} \in \mathbb{R}^{k\times k}$ can be *lifted*[1] to the matrix $A^{\uparrow}_{\mathcal{S}} \in \mathbb{R}^{n\times n}$ defined by

$$A^{\uparrow}_{\mathcal{S}}(v, w) = A_{\mathcal{S}}(s(v), s(w)).$$

Given a k-partition $\mathcal{P} = \{S_1, \ldots, S_k\}$ of $[n]$, we say that a symmetric $n \times n$ matrix M with real entries is $\mathcal{P}$-*constant* if the $S_i \times S_j$ submatrix of M is constant, $1 \leq i, j \leq k$. More formally, M is $\mathcal{P}$-constant if for all pairs (i, j), $1 \leq i, j \leq k$, there is a constant $c_{ij} = c_{ji}$ such that $M(p, q) = c_{ij}$ for each

[1]Our lifted matrix is slightly different from the *expected adjacency matrix* in [1], which can also be computed from our summary (see Sect. V). Our algorithms also approximate the partition that minimizes the error from the expected adjacency matrix (proof omitted for space reasons).

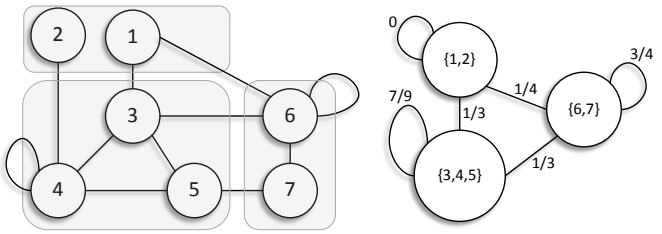

Fig. 1: *A graph G (left) and one possible summary $\mathcal{S}$ (right).*

	1	2	3	4	5	6	7
1	0	0	1/3	1/3	1/3	1/4	1/4
2	0	0	1/3	1/3	1/3	1/4	1/4
3	1/3	1/3	7/9	7/9	7/9	1/3	1/3
4	1/3	1/3	7/9	7/9	7/9	1/3	1/3
5	1/3	1/3	7/9	7/9	7/9	1/3	1/3
6	1/4	1/4	1/3	1/3	1/3	3/4	3/4
7	1/4	1/4	1/3	1/3	1/3	3/4	3/4

TABLE I: *The lifted matrix $A^{\uparrow}_{\mathcal{S}}$ corresponding to the $\mathcal{S}$ in Figure 1.*

pair (p, q) where $p \in S_i$ and $q \in S_j$. We also say that M is k-constant, to highlight the size of the partition. It should be clear from the definition that the lifted adjacency matrix of a k-summary $\mathcal{S}$ of a graph G is $\mathcal{P}_S$-constant for the partition $\mathcal{P}_\mathcal{S}$ of the nodes of G into the supernodes of $\mathcal{S}$.

An input graph, a possible summary, and the corresponding lifted matrix are exemplified in Figure 1 and Table I.

The number of possible summaries is huge (there is one for each partition of V), so we need efficient algorithms to find the summary that best resembles the graph. This goal is formalized in Problem 1, which is the focus of this work.

Problem 1 (Graph Summarization): Given a graph $G = (V, E)$ with $|V| = n$, and $k \in \mathbb{N}$, find the k-summary $\mathcal{S}^*$, such that $A^{\uparrow}_{\mathcal{S}^*}$ minimizes the error $\mathsf{err}(A_G, A^{\uparrow}_{\mathcal{S}^*})$ *for some error function* $\mathsf{err} : \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times n} \to [0, \infty)$.

The function err expresses the dissimilarity between the original adjacency matrix of G and the lifted matrix obtained from the summary. Different definitions for err are possible and different algorithms may be needed to find the optimal summary $\mathcal{S}^*$ according to different measures. In this paper we focus on the ℓ_p*-reconstruction error*, in particular we focus on ℓ_1 and ℓ_2-reconstruction errors (the algorithms in [1] try to minimize the ℓ_1-reconstruction error).

Let $p \in \mathbb{R}$, $p \geq 1$. Given a graph G with adjacency matrix A_G and a summary $\mathcal{S}$ with lifted adjacency matrix $A^{\uparrow}_{\mathcal{S}}$, the ℓ_p*-reconstruction error* of $\mathcal{S}$ is defined as the entry-wise p-norm of the difference between A_G and $A^{\uparrow}_{\mathcal{S}}$:

$$\mathsf{err}_p(A_G, A^{\uparrow}_{\mathcal{S}}) = \|A_G - A^{\uparrow}_{\mathcal{S}}\|_p = \Big(\sum_{i=1}^{|V|}\sum_{j=1}^{|V|} |A_G(i,j) - A^{\uparrow}_{\mathcal{S}}(i,j)|^p\Big)^{1/p}$$

If A_G has entries in $[0, 1]$ then $\mathsf{err}_p(A_G, A^{\uparrow}_{\mathcal{S}}) \in [0, n^{2/p}]$.

The ℓ_p norms can be computed in time $O(n^2)$ if $p = O(1)$. If the original graph $G = (V, E)$ is *unweighted*, it is possible to compute the ℓ_p-reconstruction errors in time $O(k^2)$ from the k-summary $\mathcal{S} = (V', V' \times V')$ itself. Indeed, given the partition $V' = \{V_1, \ldots, V_k\}$ of V, let $\alpha_{ij} = e_G(V_i, V_j)/(|V_i||V_j|)$ denote the superedge densities, for $i, j \in [k]$. A simple calculation shows that, for $p \geq 1$,

$$\mathsf{err}_p(A^{\uparrow}_{\mathcal{S}}, A_G)^p = \sum_{i,j\in[k]} |V_i||V_j|\alpha_{ij}(1-\alpha_{ij})\big((1-\alpha_{ij})^{p-1} + \alpha_{ij}^{p-1}\big).$$

From this we also get that, for any summary $\mathcal{S}$,

$$\mathsf{err}_2(A^{\uparrow}_{\mathcal{S}}, A_G)^2 = 2 \cdot \mathsf{err}_1(A^{\uparrow}_{\mathcal{S}}, A_G). \tag{1}$$

Thus, a partition that minimizes the ℓ_2-reconstruction error also minimizes the ℓ_1-reconstruction error. A similar statement holds approximately (up to constant factors) for the ℓ_p and ℓ_q-reconstruction errors where $1 \leq p, q \leq O(1)$, so for unweighted graphs the exact choice of p is not crucial.

III. Summarization with guarantees

We now show a close connection between graph summarization and well-studied geometric clustering problems (k-median/means).

The best matrix for a given partition. Given $p \geq 1$, a graph $G = (V, E)$ (w.l.o.g. $V = [n]$), and a partition $\mathcal{P} = \{S_1, \ldots, S_k\}$ of $[n]$, what $\mathcal{P}$-constant matrix $B^{*,p}_{\mathcal{P}}$ minimizes $\left\|A_G - B^{*,p}_{\mathcal{P}}\right\|_p$? We now justify the use of the ($\mathcal{P}$-constant) lifted adjacency matrix in our analysis.

Clearly it suffices to consider each pair of supernodes S_i, S_j separately. For a fixed pair, let X be a random variable representing the weight (in G) of a edge (x, y) drawn uniformly at random from $S_i \times S_j$. We are looking for the real number a_p that minimizes $\mathbb{E}[|X - a_p|^p]$; this is known as the *p-predictor* of X. Therefore, $B^{*,p}_{\mathcal{P}}(x, y)$ is equal to the p-predictor of the uniform distribution over the multiset $M(x, y) = \{A_G(v, w) \mid v \in s(x), y \in s(y)\}$, where $s(v)$ denotes the unique set of $\mathcal{P}$ to which v belongs. It is well-known that the 1-predictor of X is its median, and its 2-predictor is its expectation. In other words,

$$B^{*,1}_{\mathcal{P}}(x, y) = \text{median}(\{A_G(v, w) \mid (v, w) \in s(v) \times s(w)\}),$$
$$B^{*,2}_{\mathcal{P}}(x, y) = \sum_{(v,w)\in s(v)\times s(w)} A_G(v, w)/(|s(x)||s(y)|).$$

Note that $B^{*,2}_{\mathcal{P}} = A^{\uparrow}_{\mathcal{S}_\mathcal{P}}$, the lifted adjacency matrix of the summary $\mathcal{S}_\mathcal{P}$ corresponding to $\mathcal{P}$, which in general is different from $B^{*,1}_{\mathcal{P}}$. However, $A^{\uparrow}_{\mathcal{S}_\mathcal{P}}$ has the advantage of being easier to handle analytically, and also provides a good approximation to $B^{*,1}_{\mathcal{P}}$, as shown in the following lemma, which is a corollary of Lemma 2.

Lemma 1:

$$\left\|A_G - B^{*,1}_{\mathcal{P}}\right\|_1 \leq \left\|A_G - B^{*,2}_{\mathcal{P}}\right\|_1 \leq 2 \cdot \left\|A_G - B^{*,1}_{\mathcal{P}}\right\|_1$$
$$\left\|A_G - B^{*,2}_{\mathcal{P}}\right\|_2 \leq \left\|A_G - B^{*,1}_{\mathcal{P}}\right\|_2 \leq \sqrt{2} \cdot \left\|A_G - B^{*,2}_{\mathcal{P}}\right\|_2.$$

Lemma 2: Let X be a random variable with median m and expectation μ. Then

$$\mathbb{E}[|X - m|] \leq \mathbb{E}[|X - \mu|] \leq 2 \cdot \mathbb{E}[|X - m|], \tag{2}$$
$$\mathbb{E}[|X - \mu|^2] \leq \mathbb{E}[|X - m|^2] \leq 2 \cdot \mathbb{E}[|X - \mu|^2].$$

Proof: The first inequality of each line follows from the fact that m is the 1-predictor and μ the 2-predictor.

Now we bound the deviation between mean and median. Observe that

$$\begin{aligned}|\mu - m| &= |\mathbb{E}[X] - m| = |\mathbb{E}[X - m]| \\ &\leq \mathbb{E}[|X - m|] \leq \mathbb{E}[|X - \mu|] \leq \sigma,\end{aligned}$$

where $\sigma = \sqrt{\mathrm{Var}X}$ is the standard deviation of X and the last inequality is Cauchy-Schwarz.

This yields the other two inequalities:

$$\begin{aligned}\mathbb{E}[|X - \mu|] &= \mathbb{E}[|X - m + m - \mu|] \\ &\leq \mathbb{E}[|X - m|] + |m - \mu| \leq 2 \cdot \mathbb{E}[|X - m|],\end{aligned}$$

and, since $\mathbb{E}[|X - \mu|^2] = \mathrm{Var}[X] = \sigma^2$,
$E[|X - m|^2] = \mathrm{Var}[X] + (m - \mu)^2 \leq 2 \cdot \mathbb{E}[|X - \mu|]^2$. ∎

Connection with ℓ_p^p clustering. In the ℓ_p^p clustering problem, we are given n points $a_1, \ldots, a_n \in \mathbb{R}^d$ and we need to find k "centroids" $c_1, \ldots, c_k \in \mathbb{R}^d$ so as to minimize $\sum_n \left\|a_i - c_{l(k)}\right\|_p^p$, where $l(i)$ is the centroid closest to a_i in the ℓ_p metric. When $p = 2$, this is the k-means problem with ℓ_2 (Euclidean) metric; when $p = 1$, this is the k-median problem with ℓ_1 metric. We consider the *continuous* version in which the centroids are allowed to be arbitrary points.

Any choice of centroids $c_1, \ldots, c_k$ gives rise to a partition $\mathcal{P}$ of $[n]$ that groups together points having the same closest centroid (assuming a scheme to break ties). Conversely, for any partition $\mathcal{P} = \{S_1, \ldots, S_k\}$ there is an optimal (i.e., minimizing ℓ_p^p cost given $\mathcal{P}$) choice $c_1^*, \ldots, c_k^*$ of centroids: c_i^* is the coordinate-wise mean of the vectors in S_i when $p = 2$, and their coordinate-wise median when $p = 1$.

We show the following connection between clustering and summarization w.r.t. the ℓ_2 and ℓ_1-reconstruction error.

Theorem 1: Let $\bar{\mathcal{S}}$ be the k-summary induced by the partition of the rows of A_G with the smallest continuous ℓ_2^2 cost, and let $\mathcal{S}^*$ be the optimal k-summary for G w.r.t. the ℓ_2-reconstruction error. The ℓ_2-reconstruction error of $\bar{\mathcal{S}}$ is a 4-approximation to the best ℓ_2-reconstruction error:

$$\mathsf{err}_2(A_G, A^{\uparrow}_{\bar{\mathcal{S}}}) \leq 4 \cdot \mathsf{err}_2(A_G, A^{\uparrow}_{\mathcal{S}^*}) \ .$$

Theorem 2: Let $\hat{\mathcal{S}}$ be the k-summary induced by the partition of the rows of A_G with the smallest continuous ℓ_1 cost, and let $\mathcal{S}^\dagger$ be the optimal k-summary for G w.r.t. the ℓ_1-reconstruction error. The ℓ_1-reconstruction error of $\hat{\mathcal{S}}$ is an 8-approximation to the best ℓ_1-reconstruction error:

$$\mathsf{err}_1(A_G, A^{\uparrow}_{\hat{\mathcal{S}}}) \leq 8 \cdot \mathsf{err}_1(A_G, A^{\uparrow}_{\mathcal{S}^\dagger}) \ .$$

Before we can prove these theorems we need some additional definitions and lemmas.

Smoothing projections and lifted matrices. Let $\mathcal{P} = \{S_1, \ldots, S_k\}$ be a partition of $[n]$ and let $\mathbf{s}_i$ be the n-dimensional vector associated to S_i such that the j^{th} entry of $\mathbf{s}_i$ is 1 if $j \in S_i$, and 0 otherwise. Write $\mathbf{v}_i = \mathbf{s}_i / \sqrt{|S_i|}$. Since $\|\mathbf{v}_i\| = 1$ and $S_i \cap S_j = \emptyset$ for $i \neq j$, the vectors $\{\mathbf{v}_i\}_{i \in [k]}$ are orthonormal. A sequence of vectors $\mathbf{v}_1, \ldots, \mathbf{v}_k \in \mathbb{R}^n$ is *partition-based* if they arise in this way from a partition of $[n]$. We say that a linear operator $P : \mathbb{R}^n \to \mathbb{R}^n$ is *smoothing* if it can be written as $P = \sum_{i=1}^k \mathbf{v}_i \mathbf{v}_i^\intercal$ for a partition-based set of vectors $\mathbf{v}_1, \ldots, \mathbf{v}_k$. Since $P^2 = P$, $P^\intercal = P$ and $P\mathbf{v}_i = \mathbf{v}_i$, it follows that P is the orthogonal projection onto the subspace generated by $\mathbf{v}_1, \ldots, \mathbf{v}_k$. It is also easy to check that a $n \times n$ matrix A is $\mathcal{P}$-constant if and only if $PAP = A$.

Given a k-summary $\mathcal{S}$ of G, let $\mathcal{P}_{\mathcal{S}} = \{S_1, \ldots, S_k\}$ be the partition of $[n]$ corresponding to $\mathcal{S}$. Consider the smoothing projection P arising from $\mathcal{P}_{\mathcal{S}}$ as described above.

Lemma 3: $A^{\uparrow}_{\mathcal{S}} = PA_GP$.

Proof: Let $\mathbf{v}_1, \ldots, \mathbf{v}_k$ be the partition-based vectors arising from $\mathcal{P}_{\mathcal{S}}$. Recall that the entry $A^{\uparrow}_{\mathcal{S}_{\mathcal{P}}}(p, q)$ of the lifted adjacency matrix $A^{\uparrow}_{\mathcal{S}_{\mathcal{P}}}$, where $p \in S_i$ and $q \in S_j$, equals the density $d_G(S_i, S_j) = \mathbf{s}_i^\intercal A_G \mathbf{s}_j / (|S_i||S_j|)$. Therefore

$$\begin{aligned}A^{\uparrow}_{\mathcal{S}} &= \sum_{i,j \in [k]} d_G(S_i, S_j) \mathbf{s}_i^\intercal \mathbf{s}_j = \sum_{i,j \in [k]} \frac{\mathbf{s}_i^\intercal A_G \mathbf{s}_j}{|S_i||S_j|} \mathbf{s}_i^\intercal \mathbf{s}_j \\ &= \sum_{i,j \in [k]} (\mathbf{v}_i^\intercal A_G \mathbf{v}_j) \mathbf{v}_i^\intercal \mathbf{v}_j = \sum_{i,j \in [k]} \mathbf{v}_i^\intercal (\mathbf{v}_i^\intercal A_G \mathbf{v}_j) \mathbf{v}_j = PA_GP \ .\end{aligned}$$

∎

To prove Thm. 1 and Thm. 2 we also make use of the following technical lemmas.

Lemma 4: Let $P \colon \mathbb{R}^n \to \mathbb{R}^n$ be an orthogonal projection and let $\|\cdot\|$ denote a matrix norm that is (1) invariant under transposition and negation ($\|X\| = \|-X\| = \|X^\intercal\|$); and (2) contractive under P (for any $n \times n$ matrix X, $\|XP\| \leq \|X\|$). Then for any symmetric or skew-symmetric matrix A, it holds that

$$\frac{\|A - AP\|}{2} \leq \|A - PAP\| \leq 2 \|A - AP\| \, .$$

Proof: Using that $P^2 = P$ and the triangle inequality for $\|\cdot\|$, we compute

$$\begin{aligned}&\|A - AP\| = \|A - PAP + PAP - AP\| \\ &\leq \|A - PAP\| + \|PAP - AP\| = \|A - PAP\| + \|(PAP - A)P\| \\ &\leq \|A - PAP\| + \|PAP - A\| = 2 \|A - PAP\| \, .\end{aligned}$$

Observe that if A is symmetric ($A^\intercal = A$), then $(A - AP)^\intercal = A^\intercal - P^\intercal A^\intercal = A - PA$, whereas if A is skew-symmetric ($A^\intercal = -A$), $(A - AP)^\intercal = -(A - PA)$; either way, $\|A - AP\| = \|A - PA\|$. Therefore

$$\begin{aligned}&\|A - PAP\| = \|A - AP + AP - PAP\| \\ &= \|A - AP\| + \|(A - PA)P\| \leq \|A - AP\| + \|A - PA\| \\ &= \|A - AP\| + \|A - AP\| = 2 \|A - AP\| \, .\end{aligned}$$

∎

Lemma 5: The ℓ_p norms ($p \geq 1$) satisfy the conditions of Lemma 4 for any smoothing projection P.

Proof: Invariance under transposition and negation is trivial, so we only need to check the second condition. To see it, write X by columns: $X = (x_1 \mid \cdots \mid x_n)$. Then $PX = (Px_1 \mid \cdots \mid Px_n)$ and $\|XP\|_p^p = \|PX\|_p^p = \sum_i \|Px_i\|_p^p$, so it suffices to show that $\|Py\|_p \leq \|y\|_p$ for $p = 1, 2$ and all $y \in \mathbb{R}^n$. The reader can verify that this follows from the power mean inequality: $\left(\frac{|\sum_{i=1}^m y_i|}{m}\right)^p \leq \frac{\sum_{i=1}^m |y_i|^p}{m}$. ∎

The following result is an easy consequence of the definitions of smoothing projection and cost of a clustering.

Lemma 6: The ℓ_2^2 cost of the clustering associated with a partition $\mathcal{P}$ of the rows of a matrix $A \in \mathbb{R}^{n\times n}$ is $\|A - AP\|_2^2$, where P is the smoothing projection arising from $\mathcal{P}$.

We are now ready to prove Thm. 1 and 2, showing the connection between summarization and geometric clustering.

Proof of Thm. 1: Let P denote a smoothing projection arising from an arbitrary k-partition. Let $P_{\bar{\mathcal{S}}}$ be the smoothing projection induced by $\bar{\mathcal{S}}$. By Lemma 6, $\|A_G - A_G P_{\bar{\mathcal{S}}}\|_2 \leq \|A_G - A_G P\|_2$. Let $P_{\mathcal{S}^*}$ be the smoothing projection associated with the partition which minimizes the ℓ_2-reconstruction error (i.e., the one induced by $\mathcal{S}^*$). Using Lemmas 4 and 5,

$$\begin{aligned}\mathsf{err}_2(A_G, A^{\uparrow}_{\bar{\mathcal{S}}}) &= \|A_G - P_{\bar{\mathcal{S}}} A_G P_{\bar{\mathcal{S}}}\|_2 \leq 2\,\|A_G - A_G P_{\bar{\mathcal{S}}}\|_2 \\ &\leq 2\,\|A_G - A_G P_{\mathcal{S}^*}\|_2 \leq 4\,\|A_G - P_{\mathcal{S}^*} A_G P_{\mathcal{S}^*}\|_2 \\ &= 4 \cdot \mathsf{err}_2(A_G, A^{\uparrow}_{\mathcal{S}^*}).\end{aligned}$$ ∎

The proof for Thm. 2 follows the same steps but Lemma 1 must be taken into account, which results in an additional factor 2 in the approximation guarantee.

IV. An efficient algorithm for summarization

Theorems 1 and 2) showed that building a graph summary of guaranteed quality with regard to the ℓ_1 or ℓ_2-reconstruction error can be approximately reduced to solving a clustering instance. We now turn our attention to how to do this efficiently.

Both k-median and k-means are NP-hard, but admit constant-factor approximation algorithms that run in time polynomial in the number of points (n), clusters (k), and the dimension of the space [4], [5]. In order to use these algorithms for our purposes, we need to take care of the following bottlenecks: costly pairwise distances computation, high dimensionality, and high number of points. Exact computation of all pairwise distances between the rows of the adjacency matrix can be rather expensive: in the ℓ_2 norm, computing all distances between n points in $\mathbb{R}^d$ is equivalent to multiplying a matrix with its transpose.[2] We can avoid this by using approximate distances computed efficiently from a *sketch* of the adjacency matrix, i.e., a matrix with the same number of rows but a logarithmic number of columns [6]; this also reduces the number of dimensions from n to $O(\log n)$. In exchange for this speedup, the analysis needs to factor in the additional error, and the fact that the approximate distances we work with will not satisfy the triangle inequality, whereas the clustering algorithms we use are designed for metric spaces.

The $O(1)$-approximation algorithm from [7] can be used with the approximate distances computed from the sketch, but it runs in time $\widetilde{O}(n^2)$. To improve this we use a result from [8] which adaptively selects $O(k)$ of the rows of the sketch so that the optimal k-median/means solution obtained by clustering these rows gives a set of centers that can be used to obtain a constant-factor approximation to the clustering problem for all the rows. By running the algorithm from [7] on the resulting $O(k) \times O(\log n)$ matrix, we obtain a constant factor approximation in time $\widetilde{O}(m + nk)$, where m is the number of edges (or half the number of non-zero entries in A_G, if G is weighted). We formalize this intuition next.

Theorem 3: Let $p \in \{1, 2\}$. Algorithm 1 computes an $O(1)$-approximation to the best k-summary under the ℓ_p-reconstruction error in time $\widetilde{O}(m + nk)$ with high constant probability.

Proof sketch: Using techniques from [6] we can construct an $n \times O(\log n)$ linear sketch in time $\widetilde{O}(n)$, and apply it to all rows of the adjacency matrix A_G in time $\widetilde{O}(m + n)$, obtaining an $n \times O(\log n)$ sketched matrix S. Afterwards we can approximately answer any query about the distance between two rows of A_G in time $O(\log n)$, and with high probability all estimates are to within a constant factor of the actual value. The p-th powers of the pairwise row distances computed from S yield what is known as an $O(1)$-approximate semimetric with distance ratio $\mathrm{poly}(n)$, which enables us to use the metric k-median algorithms detailed below.

We use a result from [8] to select $O(k)$ rows from S in time $\widetilde{O}(nk)$. With high constant probability, the selected rows can be used to obtain (in $\widetilde{O}(nk)$) an $O(1)$-approximation to the optimal clustering of all the rows of S. We can then use the algorithm from [7] to obtain a $O(1)$-approximation to the clustering of the $O(k)$ rows of S. The *optimal* clustering of the rows of S is also a $O(1)$-approximation to the optimal clustering of the rows in the original adjacency matrix, so a $O(1)$-approximation to the former is a $O(1)$-approximation to the latter. Given this partition, we can then compute the densities in time $O(m + k^2) = O(m + nk)$. ∎

Algorithm 1: Graph summarization with ℓ_p-reconstr. error

Input : $G = (V, E)$ with $|V| = n$, $k \in \mathbb{N}$, $p \in \{1, 2\}$
Output: A $O(1)$-approximation to the best k-summary for G under the ℓ_p-reconstruction error

// Create the $n \times O(\log n)$ sketch matrix [6]
$S \leftarrow$ `createSketch` $(A_G, O(\log n), p)$
// Select $O(k)$ rows from the sketch [8]
$R \leftarrow$ `reduceClustInstance` (A_G, S, k)
// Run the approximation algorithm [7] to obtain a partition.
$\mathcal{P} \leftarrow$ `getApproxClustPartition` (p, k, R, S)
// Compute the densities for the summary
$D \leftarrow$ `computeDensities` $(\mathcal{P}, A_G)$
return $(\mathcal{P}, D)$

V. Query answering

Following [1] we adopt an *expected-value semantics* for approximate query answering: the answer to a query on the summary is the expectation of the exact answer over all graphs that may have resulted in that summary, considered all equally likely under the principle of indifference. In particular, LeFevre and Terzi [1] define an *expected adjacency matrix* $\bar{A}$ which is slightly different from the lifted matrix $A^{\uparrow}_{\mathcal{S}}$ we defined in Sect. II but can be computed from it as follows[3]:

[2] If $v_1, \ldots, v_n \in \mathbb{R}^d$, then $\|v_i - v_j\|_2^2 = \|v_i\|_2^2 + \|v_j\|_2^2 - 2\langle v_i, v_j\rangle$. Since the quantities $\|v_i\|_2^2$ can be easily precomputed, the problem reduces to computing all inner products $\langle v_i, v_j\rangle$. These form the entries of $AA^{\intercal}$, where A is the $n \times d$ matrix with rows $v_1, \ldots, v_n$.

[3] Minor modifications are needed if self-loops are allowed.

- If two vertices i and j belong to different supernodes in the summary, then $\bar{A}(i,j) = A^{\uparrow}_{\mathcal{S}}(i,j)$.
- If i and j belong to the same supernode S_ℓ, and $i \neq j$, then $\bar{A}(i,j) = A^{\uparrow}_{\mathcal{S}} \cdot |S_\ell|/(|S_\ell| - 1)$.
- If $i = j$, then $\bar{A}(i,j) = 0$.

Under the expected-value semantics, computing the answers to many important class of queries is straightforward. For instance, the *existence probability of an edge* (u,v) (or is expected weight, in case of weighted graphs) is $\bar{A}(u,v)$. The *weighted degree* of v is $\sum_{i=1}^{n} \bar{A}(v,i)$. Similarly, the *weighted eigenvector centrality* can be expressed as $\sum_{i=1}^{n} \bar{A}(v,i)/2|E|$.

It is worth remarking that the average error of adjacency queries is the ℓ_1-reconstruction error, while the average error of degree queries is always bounded by the ℓ_1-reconstruction error divided by n. Hence in these cases it is easy to prove worst-case bounds on the average error incurred when computing the answer from the summary.

We next show how to answer queries involving the number of triangles.

Let n_i be the number of vertices in the i-th supernode and let π_{ij} be defined as follows for $1 \leq i,j \leq k$: $\pi_{ij} = d_{ij}$ if $i \neq j$, and $\pi_{ij} = \frac{d_{ij} n_i}{n_i - 1}$ if $i = j$.

The following result follows from linearity of expectation.

Lemma 7: The expected number of triangles is

$$\mathbb{E}[\triangle] = \sum_{i=1}^{k} \left(\binom{n_i}{3} \pi_{ii}^3 + \sum_{j=i+1}^{k} \left(\pi_{ij}^2 \left(\binom{n_i}{2} n_j \pi_{ii} + \binom{n_j}{2} n_i \pi_{jj} \right) + \sum_{w=j+1}^{k} n_i n_j n_w \pi_{ij} \pi_{jw} \pi_{wj} \right) \right). \quad (3)$$

It can be computed in time $O(k^3)$.

The same approach can be used to develop formulas for the expected distribution of subgraphs of any size. Care must be taken to avoid counting the same occurrence of a subgraph multiple times due to isomorphisms.

The *triangle density* of a graph is the ratio between the number of triangles in the graph over the number of triplets of vertices, independently of their connectivity. The results above allow us also to compute the expected triangle density from the summary.

Corollary 1: Let $\mathbb{E}[\triangle]$ be the expected number of triangles from (3)*. Then the expected triangle density is*

$$\frac{6\mathbb{E}[\triangle]}{n(n-1)(n-2)} \ . \quad (4)$$

VI. Experimental Evaluation

In this section we report the results of our experimental evaluation which has the following goals: *(1)* to characterize the structure of the summaries built by our algorithms; *(2)* to evaluate the quality of the summaries in terms of the reconstruction errors and the cut-norm error and of their usefulness in answering queries; *(3)* to compare the performances of our algorithms with those of `GraSS` from [1].

Datasets and implementations. We used real graphs from the SNAP repository[4]. As the considered graphs are unweighted, the ℓ_1-reconstruction error is half the *squared* ℓ_2-reconstruction error (see (1)), hence we only report the results for the ℓ_2-error (divided by n for normalization).

We consider two variants of our method based on different variants of the k-median clustering procedure: "`S2A`" is the algorithm for the ℓ_2-reconstruction error using the constant-factor approximation algorithm from [5], while "`S2L`" uses the classic Lloyd's iterative approach [9] with `k-means++` initialization that guarantees an $O(\log k)$ approximation factor [10]. Our algorithms are implemented[5] in C++11 and the experiments are performed on a 4-core AMD Phenom II X4 955 with 16GB of RAM running GNU/Linux 3.12.9. Each algorithm is run 5 times for each combination of parameters.

Summary characterization. We studied the structure of the summaries created by our algorithms in terms of the distribution of the sizes of the supernodes, the distributions of the internal and cross densities, the (reconstruction or cut-norm) error of the generated summaries, and the running time of our algorithms. As expected "`S2A`" is slower but more accurate than `S2L`: due to space limitations we only report results for `S2L` in Table II. We do not report the minimum size since this was always 1 in all cases. This is interesting: in order to minimize the ℓ_2-reconstruction error it may actually be convenient to create a supernode containing a single vertex. Nevertheless there are also large supernodes containing hundreds or thousands of vertices, which helps explain the relatively large standard deviation. As k grows, the standard deviation shrinks faster than the average size (n/k), suggesting that supernode sizes become more uniform.

The minimum internal density was 0 in all our tests, as a consequence of aforementioned fact that there are supernodes of size 1 and that the graphs had no self-loops. On the other hand, there are supernodes whose corresponding induced subgraphs are quite dense, almost cliques (a clique would correspond to a value of 100 in the "max" column). The minimum and maximum cross densities are not reported because they were respectively 0 and 1 in all cases. While the latter fact is expected from the presence of supernodes of size 1, the former suggests that some supernodes are effectively *independent* from each other, i.e., there are no edges connecting them. Finally, as expected, ℓ_2-reconstruction error shrinks linearly and running time grows linearly as k grows.

Query answering. In Table III we report *(1)* the absolute error for adjacency queries, *(2)* the absolute degree error, and *(3)* the *relative* triangle density error. Results for the very large graphs are not available because computing the query error would require running the query on the original graph, and this takes an excessive amount of time (indeed, this is one of the motivations for our work). In general, as expected, a decrease in k corresponds to an often-substantial increase in the query answer error. For adjacency queries, the average error (which is exactly the ℓ_1-reconstruction error) is very

[4] http://snap.stanford.edu/data/

[5] The implementations and datasets are available at https://db.tt/7YXGDqbs .

Graph	k	Size stdev	Size max	Internal Density ($\times 10^2$) avg	Internal Density ($\times 10^2$) stdev	Internal Density ($\times 10^2$) max	Cross Density ($\times 10^2$) avg	Cross Density ($\times 10^2$) stdev	ℓ_2-rec. err. ($\times 10^2$) avg	Time (s) avg	Time (s) stdev
Facebook	1000	15.84	597	28.59	31.22	94.79	1.56	11.42	5.81	2.67	0.02
$\|V\| = 4\,039$	1250	11.09	382	23.52	29.64	95.15	1.44	11.18	5.42	3.53	0.01
$\|E\| = 88\,234$	1500	8.77	206	19.72	28.02	94.18	1.37	11.07	5.01	4.48	0.01
Enron	10000	42.10	4041	12.58	24.52	87.5	0.1	3.27	0.72	253.1	15.66
$\|V\| = 36\,692$	12000	27.57	2635	11.16	23.24	88.88	0.08	2.86	0.63	305.38	20.67
$\|E\| = 183\,831$	14000	23.98	2398	9.77	21.77	87.5	0.06	2.54	0.54	349.31	17.25
Stanford	2000	2481.49	113572	28.05	32.57	97.95	0.08	2.73	0.48	389.57	19.10
$\|V\| = 281\,903$	6000	1007.47	83444	24.25	31.52	97.61	0.04	1.98	0.42	970.74	9.04
$\|E\| = 1\,992\,636$	10000	658.67	65659	21.32	30.63	97.61	0.03	1.70	0.38	1604.85	81.47
Amazon0601	2000	7479.31	351920	37.79	28.91	90.9	0.01	0.9	0.53	1921.29	76.42
$\|V\| = 403\,394$	6000	3766.88	306673	36.97	29.69	90.9	0	0.76	0.52	3419.72	76.94
$\|E\| = 2\,443\,408$	8000	3053.54	278468	36.78	29.99	90.9	0	0.73	0.51	4215.82	33.32

TABLE II: *Supernode size, internal and cross densities, normalized ℓ_2-reconstruction error, and runtime for summaries built with* `S2L`.

Graph	k	Error in Query Answering: Adjacency ($\times 10^2$) avg	Adjacency ($\times 10^2$) stdev	Degree avg	Degree stdev	Triangle Density
Facebook	500	0.42	4.57	7.14	10.43	-0.31
	750	0.37	4.32	6.22	9.15	-0.28
	1000	0.33	4.05	5.38	7.96	-0.24
	1250	0.28	3.79	4.79	7.27	-0.19
	1500	0.24	3.49	4.01	6.31	-0.15
Enron	4000	0.01	0.78	2.57	4.95	-0.32
	6000	< 0.01	0.66	1.92	3.53	-0.20
	8000	< 0.01	0.57	1.49	2.65	-0.13

TABLE III: *Error in query answering for summaries built with* `S2L`. *For adjacency and degree queries we report the absolute error, while for triangle density we report the relative error.*

k	Alg.	c (for `GS`)	ℓ_2-reconstr. Err.	Runtime (s)
10	`GS`	0.1	0.168	495.122
		0.5	0.155	2669.961
		1.0	0.153	5516.915
	`S2A`		0.152	0.440
50	`GS`	0.1	0.146	495.518
		0.5	0.136	2671.848
		1.0	0.133	5527.319
	`S2A`		0.131	0.695
100	`GS`	0.1	0.130	495.074
		0.5	0.120	2669.013
		1.0	0.116	5508.125
	`S2A`		0.115	0.708

TABLE IV: *Comparison between* `S2A` *and* `GS` *on a random sample ($n = 500$) of* `ego-gplus` *(averages over five runs).*

small, almost 0, and indeed the error was 0 for many pairs of vertices. We found though that the maximum error could be large in some rare case. This has obviously an impact on the standard deviation that is substantially larger than the average. For degree queries, the average error is small, when compared to the average degree $2|E|/|V|$ and to the ratio between the ℓ_1-reconstruction error and n (which is an upper bound to the average degree error). As for triangle density, the estimations obtained from the summary are of good quality when the ratio between the number of vertices in the graph and the number of supernodes is not too large, but grows rapidly otherwise. This is to be expected: the number of triangles is particularly sensitive to loss of information due to summarization. Note that we always underestimate the triangle density because real-world networks have many more triangles than random graphs.

Comparison with GraSS. We compared the runtime and the summary quality of `S2A` with those of the GraSS `k-GS-SamplePairs` algorithm from [1] (which we refer to as "`GS`"), which was originally presented as a method to build summaries by heuristically minimizing the ℓ_1-reconstruction error using the *expected adjacency matrix*, rather than the *lifted matrix* from the summary. Given the close similarity between the two, we adapted `GS` to use the lifted matrix and extended it to minimize the ℓ_2-reconstruction error. In order to keep the running time of `GS` within reasonable limits, we used sampled-down versions of the graphs obtained with a "forest-fire" sampling approach. In Table IV we report the results for a sample of 500 vertices and 3969 edges of the `ego-gplus` networks. Note that `GS` takes a parameter c to quantify the number of sampled pair candidates for merging per step. We used $c \in \{0.10, 0.5, 1.0\}$. We did not use higher values for c due to the excessive running time of `GS` for high values of this parameter. `S2A` is several orders of magnitude faster than `GS` (which runs in $O(n^4 \cdot c)$), and its error is always smaller.

VII. Conclusions and Acknowledgments

This work provides the first polynomial-time approximation algorithm for the graph summarization problem defined in [1]. Our algorithm exploits a novel connection between graph summarization and the k-median and k-means problems.

The work was done during an internship of Matteo Riondato at Yahoo Labs Barcelona, while he was a Ph.D. student at Brown University, supported in part by grant NSF BIGDATA Award IIS 1247581.

References

[1] K. LeFevre and E. Terzi, "GraSS: Graph structure summarization," in *SDM*. SIAM, 2010, pp. 454–465.
[2] S. Navlakha, R. Rastogi, and N. Shrivastava, "Graph summarization with bounded error," in *SIGMOD*, 2008, pp. 419–432.
[3] J. H. Ward, "Hierarchical grouping to optimize an objective function," *J. Amer. Statist. Assoc.*, vol. 58, no. 301, pp. 236–244, 1963.
[4] K. Jain and V. V. Vazirani, "Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and Lagrangian relaxation," *J. ACM*, vol. 48, no. 2, pp. 274–296, 2001.
[5] V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pandit, "Local search heuristics for k-median and facility location problems," *SIAM J. Comput.*, vol. 33, no. 3, pp. 544–562, 2004.
[6] P. Indyk, "Stable distributions, pseudorandom generators, embeddings, and data stream computation," *J. ACM*, vol. 53, no. 3, 2006.
[7] R. R. Mettu and C. G. Plaxton, "The online median problem," *SIAM J. Comput.*, vol. 32, no. 3, pp. 816–832, 2003.
[8] A. Aggarwal, A. Deshpande, and R. Kannan, "Adaptive sampling for k-means clustering," in *APPROX-RANDOM*, 2009, pp. 15–28.
[9] S. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inf. Theor.*, vol. 28, no. 2, pp. 129–137, 1982.
[10] D. Arthur and S. Vassilvitskii, "k-means++: the advantages of careful seeding," in *Proc. of 18th SODA*, 2007, pp. 1027–1035.