2018 IEEE 59th Annual Symposium on Foundations of Computer Science

Metric Sublinear Algorithms via Linear Sampling

Hossein Esfandiari
Department of Computer Science
Harvard University
Cambridge, MA
esfandiari@seas.harvard.edu

Michael Mitzenmacher
Department of Computer Science
Harvard University
Cambridge, MA
michaelm@eecs.harvard.edu

***Abstract*—We provide a new technique to design fast approximation algorithms for graph problems where the points of the graph lie in a metric space. Specifically, we present a sampling approach for such metric graphs that, using a sublinear number of edge weight queries, provides a *linear sampling*, where each edge is (roughly speaking) sampled proportionally to its weight.**

For several natural problems, such as densest subgraph and max cut, we show that by sparsifying the graph using this sampling process, we can run a suitable approximation algorithm on the sparsified graph and the result remains a good approximation for the original problem. Our results have several interesting implications, such as providing the first sublinear time approximation algorithm for densest subgraph in a metric space, and improving the running time of estimating the average distance.

I. Introduction

In this paper, we aim to design approximation algorithms for several natural graph problems, in the setting where the points in the graph lie in a metric space. Following the seminal work of [20], we aim to provide *sublinear* approximation algorithms; that is, on problems with n points and hence $\binom{n}{2}$ edge distances, we aim to provide randomized algorithms that require $o(n^2)$ time and in fact only consider $o(n^2)$ edges, by making use of sampling. Similar to the previous work, we assume we can query the weight of any single edge in $O(1)$ time; when we use the term "query", we mean an edge weight query throughout.

A well known technique to design sublinear algorithms is uniform sampling; that is, a subset of edges (or vertices) is sampled uniformly at random. Several algorithms use uniform sampling to improve speed, space, or the number of queries [1], [5], [6], [9], [14], [17], [25]. Uniform sampling is very easy to implement, but problematically it is oblivious to the edge weights. When it comes to maximization problems on graphs, a few high weight edges may have a large effect on the solution, and hence the uniform sampling technique may fail to provide a suitable solution because it fails to sample these edges. For example, consider the densest subgraph problem, where the density of a subgraph is the sum of the edges weights divided by the number of vertices. It is known that for general unweighted graphs, the densest subgraph of a uniformly sampled subgraph with $\tilde{O}(\frac{n}{\epsilon^2})$ edges is a $1-\epsilon$ approximation of the densest subgraph of the original graph [17], [25], [26]. However, this result is not true for weighted graphs, even in a metric space (see the arxiv version [18] for an example). This problem suggests we should design approaches that sample edges with probabilities proportional to (or otherwise related to) their weight in a metric space.

As our main result, we design a novel sampling approach using a sublinear number of queries for graphs in a metric space, where independently for each edge, the probability the edge is in the sample is proportional to its weight; we call such a sampling a *linear sampling*. Specifically, for a fixed factor α, we can ensure for an edge e with weight w_e, if $\alpha w_e \leq 1$ then the edge appears in the sample with weight 1 with probability αw_e, and if $\alpha w_e > 1$, then the edge is in the sample with weight αw_e. Hence the edge weights are "downsampled" by a factor of α, in a natural way. We can choose an α to suitably sparsify our sample, graph, run an approximation algorithm on that sample, and use that result to obtain a corresponding, nearly-as-good approximation to the original problem. Interestingly, we only query $\tilde{O}(n+\beta)$[1] edge weights to provide the sample, where β is "almost" the expected weight of the edges in the sampled graph. (See Subsection I-A for a formal definition). Our algorithm to construct the sample also runs in $\tilde{O}(n+\beta)$

First author is supported in part by NSF grants CCF-1320231 and CNS-1228598.

Second author is supported in part by NSF grants CCF-1563710, CCF-1535795, CCF-1320231, and CNS-1228598. Part of this work was done while visiting Microsoft Research New England.

[1] $\tilde{O}(\cdot)$ notation hides logarithmic factors.

2575-8454/18/$31.00 ©2018 IEEE
DOI 10.1109/FOCS.2018.00011

time.

Utilizing our sampling approach, we show that for several problems a ϕ-approximate solution on a linear sample with expected weight (roughly) $\beta \in o(n^2)$ is a $(\phi - \epsilon)$-approximate solution on the input graph. From an information theory perspective this says that $\tilde{O}(n+\beta)$ queries are sufficient to find a $1-\epsilon$ approximate solution for these problems. Moreover, as the sampled graph has a reduced number of edges, if an approximation algorithm on the sampled graph runs in linear time on the sampled edges, the total time is sublinear in the size of the original graph.

In what follows, after describing the related work and a summary of our results, we present our sampling method. Our approach decomposes the graph into a sequence of subgraphs, where the decomposition depends strongly on the fact that the graph lies in a metric space. Using this decomposition, and an estimate of the average edge weight in the graph, we can determine a suitable sampled graph. We then show this sampling approach allows us to find sublinear approximation algorithms for several problems, including densest subgraph and max cut, in the manner described above. Because of space limitations, some of our results appear in the arxiv version of the paper [18].

In some applications, such as diversity maximization, it can be beneficial to go slightly beyond metric distances [33]. We can extend our results to more general spaces that satisfy what is commonly referred to as a parametrized triangle inequality [2], in which for every three points a, b and c we have $w_{a,b}+w_{b,c} \geq \lambda w_{c,a}$ for a parameter λ. As an example, if the weight of each edge (u, v) is the squared distance between the two points, the graph satisfies a parametrized triangle inequality with $\lambda = 1/2$. We provide analysis for this more general setting throughout, and refer to a graph satisfying such a parametrized triangle inequality as a λ-metric graph. (Throughout, we take $\lambda \leq 1$).

A. Our Results

As our main technical contribution we provide an approach to sample a graph $H_\beta = (V, E_H)$ from a λ-metric graph $G = (V, E_G)$ with the properties specified below that makes only $\tilde{O}(\frac{n+\beta}{\lambda})$ queries and succeeds with probability at least $1-O(1/n)$. It is easy to observe that our algorithm runs in $\tilde{O}(\frac{n+\beta}{\lambda})$ time as well.

- For some fixed factor α (which is a function of β) independently for each edge e we have:
  - If $\alpha w_e \leq 1$, we have edge e with weight 1 in E_H with probability αw_e.
  - If $\alpha w_e > 1$, we have edge e with weight αw_e in E_H.
- We have $\beta \leq \mathrm{E}\left[\sum_{e\in E_H} w'_e\right] \leq 2\beta$, where w'_e is the weight of e in H_β.[2]

As the weight of each edge in E_H is at least 1, $\mathrm{E}\left[\sum_{e\in E_H} w_e\right] \leq 2\beta$ implies that $\mathrm{E}\left[|E_H|\right] \leq 2\beta$.

To obtain our linear sampling with a sublinear number of edge queries, we cannot afford to query large numbers of low weight edges. Accordingly, our linear sampling construction utilizes a somewhat complex graph decomposition; we briefly describe some of the high-level ideas behind it. We decompose the graph into a logarithmic sequence of layers, based on the maximum edge weight. The key idea we use is that when a metric graph has a high-weight edge, say of weight W, at least one of the vertices of that edge must have at least half of its neighboring edges have weight at least $W/2$; this is the special property of metric graphs that allows our results. We can therefore find a subset of vertices that have a large number of high-weight edges (at least half the maximum edge weight) easily, with only a logarithmic number of random edge queries per vertex. We can remove the subgraph of these vertices and their neighboring edges, reduce the maximum weight of the graph by a constant factor, and continue recursively with the decomposition. This process effectively separates out a large number of low-weight edges for which $\alpha w_e \leq 1$ with a sublinear number of queries; by using a variation of rejection sampling, we can determine an appropriate sample of these low-weight edges to include gain with a sublinear number of queries.

We note that for three points a, b and c in a λ-metric space and any parameter p, $w_{a,b}+w_{b,c} \geq \lambda w_{c,a}$ directly implies $w^p_{a,b} + w^p_{b,c} \geq \frac{\lambda}{2^p} w^p_{c,a}$. Therefore one can use our technique to sample edges proportional to w^p_e (a.k.a. l_p sampling). In the streaming setting, l_p sampling has been extensively studied and appears to have several applications [28]; as far as we are aware, our approach provides the first l_p sampling techniques that uses a sublinear number of edge weight queries.

As previously mentioned, in Section III we consider several problems and show that for some $\beta \in o(n^2)$, any ϕ-approximate solution of the problem on H_β is an $(\phi - \epsilon)$-approximate solution on the original graph with high probability. Specifically, we show that $\beta \in O(\frac{n \log n}{\epsilon^2})$ is sufficient to approximate densest subgraph and max cut, $\beta \in O(\frac{n^2 \log n}{\epsilon^2 k})$ is sufficient to approximate k-hypermatching, and $\beta \in O(\frac{\log n}{\epsilon^2})$ is sufficient to approximate the average distance. Notice

[2]This can be extended to $\beta \leq \mathrm{E}\left[\sum_{e\in E_H} w'_e\right] \leq (1+\gamma)\beta$, for any arbitrary γ (See the footnote on Theorem 10 for details.) In our work, the upper bound only affects the number of queries; we prefer to set $\gamma = 1$ and simplify the argument.

that these results directly imply (potentially exponential time) $(1-\epsilon)$ approximation algorithms with sublinear number of queries for each of the problems. Often our methodology can also yield sublinear time algorithms (since it uses a sublinear number of edges) with possibly worse approximation ratios.

We now briefly describe specific results for the various problems we consider, although we defer the formal problem definitions to Section III. All of the algorithms discussed below work with high probability. We note that, throughout the paper, we use $\log n$ for $\log_e n$.

For average distance, we provide a $(1-\epsilon)$-approximation algorithm that simply finds the sum of the weights of the edges in H_β for $\beta \in O(\frac{\log n}{\epsilon^2})$, and hence our algorithm runs in time $\tilde{O}(\frac{n+\frac{1}{\epsilon^2}}{\lambda})$. For a metric graph, this improves the running time of the previous result of Indyk [20] that runs in $O(\frac{n}{\epsilon^{3.5}})$ time, with constant probability.

For densest subgraph, the greedy algorithm yields a $1/2$-approximate solution in time quasilinear in the number of edges [10]. The expected number of edges of H_β can be bounded by $\tilde{O}(\frac{n}{\lambda\epsilon^2})$ for the densest subgraph on λ-metric graphs. Therefore, our result implies a $(1/2-\epsilon)$-approximation algorithm for densest subgraph in λ-metric spaces requiring $\tilde{O}(\frac{n}{\lambda\epsilon^2})$ time.

A sublinear time algorithm for a $(1-\epsilon)$ approximation for metric max cut is already known [21]. The previous result uses $\tilde{O}(\frac{n}{\epsilon^5})$ queries, while we use only $\tilde{O}(\frac{n}{\epsilon^2})$ queries. (We note that this result does not improve the running time, but remains interesting from an information theoretic point of view. Indeed, there are several interesting results on sublinear space algorithms that ignore the computational complexity e.g., max cut [7], [23], [22], [24], set cover [3], [19], vertex cover and hypermatching [12], [11].)

Finally, on the hardness side, in Section IV we show that $\Omega(n)$ queries are necessary even if one just wants to approximate the size of the solution for densest subgraph, k-hypermatching, max cut, and average distance.

B. Other Related Work

Metric spaces are natural in their own right. For example, they represent geographic information, and hence graph problems such as the densest subgraph problem often have a natural interpretation in metric spaces. It also is often reasonable to manage large data sets by embedding objects within a suitable metric space; for example, the idea of finding network coordinates consistent with latency measurements to predict latency has been widely studied [13], [30], [31], [32].

There are several works on designing sublinear algorithms for different variants of clustering problems in metric spaces due to their application to machine learning [1], [4], [14], [15], [21]. We briefly summarize some of these papers. Alon et al. studies the efficiency of uniform sampling of vertices to check for given parameters k and b if the set of points can be clustered into k subsets each with diameter at most b, ignoring up to an ϵ fraction of the vertices [1]. Czumaj and Sohler studies the efficiency of uniform sampling of vertices for k-median, min-sum k-clustering, and balanced k-median [14]. Badoiu et al. consider the facility location problem in metric space [4]. They compute the optimal cost of the minimum facility location problem, assuming uniform costs and demands, and assuing every point can open a facility. Moreover, they show that there is no $o(n^2)$ time algorithm that approximates the optimal solution of general case of metric facility location problem to within any factor.

A basic difference between these previous works on clustering problems and the densest subgraph problem that we consider here is that all previous problems aim to decompose the graph into two or more subsets, where each subset consists of points that are close to each other. However, densest subgraph in a metric space aims to pick a diverse, spread out subset of points. (While perhaps counterintuitive, this is clear from the definition, which we provide shortly.) The application of metric densest subgraph in diversity maximization and feature selection is well studied [8], [33].

Sublinear algorithms may also refer to sublinear space algorithms such as streaming algorithms. A related, well-studied setting is semi-streaming [29], often used for graph problems. In the semi-streaming setting the input is a stream of edges and we take one (or a few) passes over the stream, while only using $\tilde{O}(n)$ space.

For the densest subgraph problem, there have been a number of recent papers showing the efficiency of uniform edge sampling in unweighted graphs [9], [17], [25], [26]. Initially, Bhattacharya et al. provided a 0.5 approximation semi-streaming algorithm for this problem [9]. They extended their approach to obtain a 0.25 approximation algorithm for this problem for dynamic streams with $\tilde{O}(1)$ update time and $\tilde{O}(n)$ space. McGregor et al. and Esfandiari et al. independently provide a $(1-\epsilon)$-approximation semi-streaming algorithm for this problem [17], [25]. Esfandiari et al. extend the analysis of uniform sampling of edges to several other problem. Mitzenmacher et al. study the efficiency of uniform edge sampling for densest subgraph in hypergraphs [26].

There are many other related problems; see [16], for example, for a survey on sublinear algorithms.

II. Providing a Linear Sampling

In this section we provide a technique to construct the desired sampled graph $H_\beta = (V, E_H)$ from a metric graph $G = (V, E_G)$. We first provide a useful decomposition of the graph. We show this decomposition allows us to obtain a graph H^α that satisfies the first property of H_β, namely that edge weights are scaled down (in expectation, for edges with scaled weights less than 1) by a factor of α. We then show how to determine a proper value α so that expected sum of the edge weight is between β and 2β as desired.

A. A Graph Decomposition

We start with a decomposition for a metric graph G, assuming an upper bound L on the weight of the edges. For an suitable number t determined later we define the following sequences.

- A sequence of graphs $G = G_1 \supseteq G_2 \supseteq \cdots \supseteq G_t$.
- A sequence of vertex sets $\nu_1, \ldots, \nu_t$.
- A sequence of weights, $L_1 = L, L_2 = \frac{L_1}{2}, \ldots, L_t = \frac{L_{t-1}}{2}$.

We denote the vertex set and edge set of G_i by V_i and E_i respectively. We begin with $G_1 = G$, and G_i is constructed from G_{i-1} by removing vertices in ν_{i-1}, i.e. $G_i = G_{i-1} \setminus \nu_{i-1}$. However, defining ν_i, which depends on G_i, requires the following additional definitions. For any $i \in \{1, \ldots, t\}$ and $\Lambda \in [0, 1]$, define $G_i^\Lambda = (V_i, E_i^\Lambda)$ to be the graph obtained by removing all edges with weight less than ΛL_i from G_i, i.e., $e \in E_i^\Lambda$ if and only if $e \in E_i$ and $w_e \geq \Lambda L_i$.

We now define G_i and ν_i iteratively as follows. We define $\nu_{i,\xi}$ to be the set of vertices in $G_i^{\lambda/4}$ with degree at least $\xi|V_i|$. We let ν_i be an arbitrary subset such that $\nu_{i,1/2} \subseteq \nu_i \subseteq \nu_{i,1/4}$. As mentioned, $G_1 = G$ and $G_i = G_{i-1} \setminus \nu_{i-1}$. We define $E_{\nu_i} = E_i \setminus E_{i+1}$. Note that E_{ν_i} is the set of edges neighboring ν_i in G_i.

Lemma 1: For any $i \in \{1, \ldots, t\}$, the set of vertices in $G_i^{\lambda/4}$ with degree at least $\frac{|V_i|}{2}$ (i.e., $\nu_{i,1/2}$) is a vertex cover for $G_i^{1/2}$.

Proof : Let $(u, v) \in E_i^{1/2}$ be an edge in $G_i^{1/2}$, and let d_u and d_v be the degrees of u and v in $G_i^{\lambda/4}$ respectively. Next we show that $d_u + d_v \geq |V_i|$. Hence we have $d_u \geq \frac{|V_i|}{2}$ or $d_v \geq \frac{|V_i|}{2}$. This means that $\nu_{i,1/2}$ covers (u, v) as desired.

Notice that, $(u, v) \in E_i^{1/2}$ means that $w_{u,v} \geq \frac{L_i}{2}$. Hence, by the λ-triangle inequality, for any $v' \in V_i$ we have $w_{(u,v')} + w_{(v',v)} \geq \lambda w_{(u,v)} \geq \frac{\lambda L_i}{2}$. Now we are ready to bound $d_u + d_v$.

$$
\begin{aligned}
&d_u + d_v \\
&= \Big(1 + \sum_{v' \in V_i \setminus \{u,v\}} 1_{(u,v') \in E_i^{\lambda/4}}\Big) \\
&\quad + \Big(1 + \sum_{v' \in V_i \setminus \{u,v\}} 1_{(v,v') \in E_i^{\lambda/4}}\Big) \qquad 1 \text{ is for } (u,v) \\
&= 2 + \sum_{v' \in V_i \setminus \{u,v\}} \Big(1_{(u,v') \in E_i^{\lambda/4}} + 1_{(v,v') \in E_i^{\lambda/4}}\Big) \\
&= 2 + \sum_{v' \in V_i \setminus \{u,v\}} \Big(1_{w_{(u,v')} \geq \lambda L_i/4} + 1_{w_{(v,v')} \geq \lambda L_i/4}\Big) \\
&\geq 2 + \sum_{v' \in V_i \setminus \{u,v\}} 1 \qquad \text{Since } w_{(u,v')} + w_{(v',v)} \geq \frac{\lambda L_i}{2} \\
&= 2 + |V_i| - 2 = |V_i|,
\end{aligned}
$$

which completes the proof. □

Lemma 2: For any $i \in \{1, \ldots, t\}$, L_i is an upper bound on weight of the edges in G_i, i.e., we have $\max_{e \in E_i} w_e \leq L_i$.

Proof : For $i = 1$, $L_1 = L$ which is an upper bound on weight of the edges in $G_1 = G$. For $i > 1$, $\nu_{i-1,1/2}$ is a vertex cover of $G_{i-1}^{1/2}$, by Lemma 1. Moreover, by definition we have $\nu_{i-1,1/2} \subseteq \nu_{i-1}$. Hence ν_{i-1} is a vertex cover of $G_{i-1}^{1/2}$. This means that every edge with weight at least $\frac{L_{i-1}}{2}$ has a neighbor in ν_{i-1}. Recall that $G_i = G_{i-1} \setminus \nu_{i-1}$, and hence, G_i has no edge with weight at least $\frac{L_{i-1}}{2} = L_i$. □

The following theorem compares the average weight of the edges in E_{ν_i} with L_i. We later use this in Theorem 7 to bound the number of queries.

Lemma 3: For any $i \in \{1, \ldots, t\}$, we have

$$\frac{\lambda}{32} L_i |V_i||\nu_i| \leq \sum_{e \in E_{\nu_i}} w_e \leq L_i |V_i||\nu_i|.$$

Proof : We start by proving the upper bound. Recall that E_{ν_i} is the set of edges neighboring ν_i in G_i. Hence the number of edges in E_{ν_i} is upper bounded by sum of the degrees of the vertices of ν_i in G_i. The degree of each vertex in G_i is $|V_i| - 1 < |V_i|$, and there are $|\nu_i|$ vertices in ν_i. Thus, we have $|E_{\nu_i}| \leq |V_i||\nu_i|$. Moreover, by Lemma 2, for each $e \in E_{\nu_i} \subseteq E_i$ we have $w_e \leq L_i$. Therefore we have

$$\sum_{e \in E_{\nu_i}} w_e \leq \sum_{e \in E_{\nu_i}} L_i \leq L_i |V_i||\nu_i|.$$

Next we prove the lower bound. Recall that we have $\nu_i \subseteq \nu_{i,1/4}$. Thus, for each $v \in \nu_i$, the degree of v in

$G_i^{\lambda/4}$ is at least $\frac{|V_i|}{4}$. Thus, for any *fixed* $v \in \nu_i$ we have

$$\begin{aligned}
&\sum_{(u,v)\in E_{\nu_i}} w_{(u,v)} \\
&= \sum_{(u,v)\in E_i} w_{(u,v)} && \text{Definition of } E_{\nu_i} \text{ for } v \in \nu_i \\
&\geq \sum_{(u,v)\in E_i^{\lambda/4}} w_{(u,v)} && G_i^{\lambda/4} \subseteq G_i \\
&\geq \sum_{(u,v)\in E_i^{\lambda/4}} \frac{\lambda L_i}{4} && \text{Definition of } G_i^{\lambda/4} \\
&\geq \frac{|V_i|}{4}\frac{\lambda L_i}{4} = \frac{\lambda}{16}|V_i|L_i. && v \in \nu_i \subseteq \nu_{i,1/4}
\end{aligned} \tag{1}$$

Note that each edge in E_{ν_i} intersects at most two vertices in ν_i. Therefore, we have

$$\begin{aligned}
\sum_{e\in E_{\nu_i}} w_e &\geq \frac{1}{2}\sum_{v\in\nu_i}\sum_{(u,v)\in E_{\nu_i}} w_{(u,v)} \\
&\geq \frac{1}{2}\sum_{v\in\nu_i}\frac{\lambda}{16}|V_i|L_i && \text{By Inequality 1} \\
&\geq \frac{\lambda}{32}L_i|V_i||\nu_i|,
\end{aligned}$$

which completes the proof of the lemma. □

Lemma 5 provides a technique to construct ν_i using $\tilde{O}(n)$ queries, with high probability. This to prove this lemma we sample some edges. Notice that these sampled edges are different from the edges that we sample to keep in H_β. We use the following standard version of the Chernoff bound (see e.g. [27]) in Lemma 5 as well as the rest of the paper.

Lemma 4 (Chernoff Bound): Let $x_1, x_2, \ldots, x_r$ be a sequence of independent 0-1 random variables, and let $X = \sum_{i=1}^r x_i$. For any $\epsilon \in [0,1]$, we have

$$\Pr\left(|X - \mathbf{E}[X]| \geq \epsilon \mathbf{E}[X]\right) \leq 2\exp(-\epsilon^2 \mathbf{E}[X]/3).$$

As we are now moving to doing sampling, we briefly remark on some noteworthy points. First, there is some probability of failure in our results. We therefore refer to the success probability in our results, and note that our algorithms may fail "silently"; that is, we may not realize the algorithm has failed (because of a low probability event in the sampling). Also, we emphasize that in general, in what follows, when referring to the number of queries required, we mean the expected number of queries. However, using expectations is for convenience; all of our results throughout the paper could instead be turned into results bounding the number of queries required with high probability (say probability $1 - O(1/n)$ using Chernoff bounds at the cost of at most constant factors in the standard way. Finally, in some places we may sample which edges we decide to query from a set of edges with a fixed probability p. In such situations, instead of iterating through each edge (which could take time quadratic in the number of vertices) we can generate the number of samples from a binomial distribution and then generate the samples without replacement; alternatively, we could determine which sample is the next sample at each step using by calculating a geometrically distributed random variable. We assume this work can be done in constant time per sample. For this reason, our time depends on the number of queries, and not the total number of edges.

Lemma 5: For any $i \in \{1, \ldots, t\}$, given G_i and L_i, one can construct ν_i using $192(\log n + \log t)|V_i| \in \tilde{O}(n)$ expected queries, succeeding with probability at least $1 - \frac{1}{nt}$.

Proof : If $|V_i| \leq 384(\log n + \log t)$, we have $E_i = \binom{|V_i|}{2} = \frac{1}{2}384(\log n + \log t)(|V_i| - 1) \leq 192(\log n + \log t)|V_i|$. Hence in this case we query all the edges and construct ν_i. In what follows we assume $|V_i| \geq 384(\log n + \log t)$. To construct ν_i we sample each edge in E_i with probability $p = \frac{384(\log n + \log t)}{|V_i|}$. We add a vertex v to ν_i if and only if at least $\frac{3}{8}384(\log n + \log t)$ of its sampled neighbors has weight $\frac{\lambda L_i}{4}$. The number of sampled edges is $p\binom{|V_i|}{2} = \frac{1}{2}384(\log n + \log t)(|V_i| - 1) \leq 192(\log n + \log t)|V_i|$.

We denote the degree of a vertex $v \in V_i$ in $G_i^{\lambda/4}$ by d_v. Let Y_e be a binary random variable that is 1 if we sample e and 0 otherwise. Let us define $Z_e = Y_e 1_{w_e \geq \lambda L_i/4}$ and $Z_v = \sum_{u\in V_i} Z_{(u,v)}$. Recall that we add $v \in V_i$ to ν_i if and only if $Z_v \geq \frac{3}{8}384(\log n + \log t)$. Notice that, for any $v \in V_i$ we have

$$\begin{aligned}
\mathrm{E}\left[Z_v\right] &= \mathrm{E}\left[\sum_{u\in V_i} Z_{(u,v)}\right] = \sum_{u\in V_i} \mathrm{E}\left[Y_{(u,v)}\right] 1_{w_{u,v}\geq\frac{\lambda L_i}{4}} \\
&= p\sum_{u\in V_i} 1_{w_{u,v}\geq\frac{\lambda L_i}{4}} = pd_v.
\end{aligned} \tag{2}$$

As Z_v is the sum of independent binary random variables, by the Chernoff bound we have

$$\begin{aligned}
&\Pr\left[|Z_v - \mathrm{E}\left[Z_v\right]| \geq \frac{384(\log n + \log t)}{8}\right] \\
&\leq 2\exp\Big(-\frac{1}{3}\big(\frac{384(\log n + \log t)}{8\,\mathrm{E}\left[Z_v\right]}\big)^2 \mathrm{E}\left[Z_v\right]\Big)
\end{aligned}$$

$$
\begin{aligned}
&= 2\exp\Big(-\frac{768(\log n+\log t)^2}{\mathrm{E}\left[Z_v\right]}\Big)\\
&= 2\exp\Big(-\frac{768(\log n+\log t)^2}{pd_v}\Big)\\
&= 2\exp\Big(-\frac{2(\log n+\log t)|V_i|}{d_v}\Big)\\
&\le 2\exp\Big(-2(\log n+\log t)\Big) && d_v \le |V_i|\\
&= \frac{2}{n^2t^2} \le \frac{1}{n^2t}. && \text{Assuming } t\ge 2
\end{aligned}
$$

By applying the union bound we have

$$
\begin{aligned}
&\Pr\left[\exists_{v\in V_i} |Z_v - \mathrm{E}\left[Z_v\right]| \ge 48(\log n+\log t)\right] \le\\
&\sum_{v\in V_i} \Pr\left[|Z_v - \mathrm{E}\left[Z_v\right]| \ge 48(\log n+\log t)\right] =\\
&|V_i|\frac{1}{n^2t} \le \frac{1}{nt}.
\end{aligned}
$$

This means that with probability at least $1-\frac{1}{nt}$, simultaneously for all vertices $v\in V_i$ we have

$$|Z_v - \mathrm{E}\left[Z_v\right]| \le 48(\log n+\log t). \tag{3}$$

Next assuming that for all vertices $v\in V_i$ we have $|Z_v - \mathrm{E}\left[Z_v\right]| \le 48(\log n+\log t)$ we show that the ν_i that we pick satisfy the property $\nu_{i,1/2}\subseteq\nu_i\subseteq\nu_{i,1/4}$.

Applying Equality 2 to Inequality 3 gives us $|Z_v - pd_v| \ge 48(\log n+\log t)$. By replacing p with $\frac{384(\log n+\log t)}{|V_i|}$ and rearranging the inequality we have

$$
\begin{aligned}
Z_v &\le \frac{384(\log n+\log t)}{|V_i|}d_v + 48(\log n+\log t)\\
&< \frac{3}{8}384(\log n+\log t) && \text{assuming } d_v < \frac{|V_i|}{4}.
\end{aligned}
$$

This means that if $d_v < \frac{|V_i|}{4}$ we have $Z_v < \frac{3}{8}384(\log n+\log t)$ and hence $v\notin\nu_i$. Therefore we have $\nu_i\subseteq\nu_{i,1/4}$. Similarly, we have

$$
\begin{aligned}
Z_v &\ge \frac{384(\log n+\log t)}{|V_i|}d_v - 48(\log n+\log t)\\
&\ge \frac{3}{8}384(\log n+\log t) && \text{assuming } d_v \ge \frac{|V_i|}{2}.
\end{aligned}
$$

This means that if $d_v \ge \frac{|V_i|}{2}$ we have $Z_v \ge \frac{3}{8}384(\log n+\log t)$ and hence $d_v\in\nu_i$. Therefore we have $\nu_{i,1/2}\subseteq\nu_i$. □

Finally, for completeness we use the following lemma to find a good upper bound L on $\max_{e\in E} w_e$ in order to start our construction of the graph decomposition (which required an upper bound on the weight of the edges).

Lemma 6: For any λ-metric graph $G=(V,E)$, one can compute a number L such that $\max_{e\in E} w_e \le L \le \frac{2}{\lambda}\max_{e\in E} w_e$ using $n-1$ queries.

Proof : Let $v'\in V$ be an arbitrary vertex. We set $L = \frac{2}{\lambda}\max_{u'\in V} w_{u',v'}$. Note that, one can simply query all the $n-1$ neighbors of v' and calculate L. Clearly, we have $L = \frac{2}{\lambda}\max_{u'\in V} w_{u',v'} \le \frac{2}{\lambda}\max_{e\in E} w_e$. Next, we show that $\max_{e\in E} w_e \le L$.

Let (u,v) be an edge such that $w_{(u,v)} = \max_{e\in E} w_e$. If $v'\in\{u,v\}$ we have $\max_{u'\in V} w_{u',v'} = \max_{e\in E} w_e$ which directly implies $L\le\frac{2}{\lambda}\max_{e\in E} w_e$ as desired. Otherwise, note that by the λ-triangle inequality we have $w_{(u,v')} + w_{(v,v')} \ge \lambda w_{(u,v)}$. Thus, we have $\max(w_{(u,v')}, w_{(v,v')}) \ge \frac{\lambda}{2}w_{(u,v)}$. Therefore, we have

$$
\begin{aligned}
L &= \frac{2}{\lambda}\max_{u'\in V} w_{u',v'} \ge \frac{2}{\lambda}\max(w_{(u,v')}, w_{(v,v')})\\
&\ge w_{(u,v)} = \max_{e\in E} w_e,
\end{aligned}
$$

as desired. □

B. Constricting H^α

We know show how to construct what we call H^α, which is derived from our original metric graph G. Recall H^α has the property that for each original edge e of weight w_e, independently, if $\alpha w_e > 1$, then H^α contains edge e with weight αw_e, and if $\alpha w_e < 1$, then H^α contains edge e with weight 1 with probability αw_e.

We define $\overline{w}$ to be the average of the weight of edges in G. We use this notion in the following lemma as well as Lemma 8 and Theorem 10.

The following theorem constructs H^α using an expected $O(n\log^2 n + n\log^2\max_{e\in E}\alpha w_e + \alpha\overline{w}\binom{n}{2})$ queries.

Theorem 7: For any α one can construct H^α using $O\big(n\log^2 n + n\log^2\max_{e\in E}\alpha w_e + \frac{1}{\lambda}\alpha\overline{w}\binom{n}{2} + \frac{n}{\lambda}\big)$ queries in expectation, succeeding with probability at least $1-\frac{1}{n}$.

Proof : By Lemma 6 we find an upper bound L on the weight of the edges, using $n-1$ queries. Recall that t is the number of graphs in our decomposition. We set $t = \log_2 n + \log_2\max_{e\in E}\alpha w_e$. Given $L_1 = L$, by definition we have

$$
\begin{aligned}
L_t &= \frac{L}{2^t}\\
&= \frac{L}{n\max_{e\in E}\alpha w_e} && t = \log_2 n + \log_2\max_{e\in E}\alpha w_e\\
&\le \frac{2}{\lambda\alpha n}. && L\le\frac{2}{\lambda}\max_{e\in E} w_e
\end{aligned}
\tag{4}
$$

Recall that, using Lemma 5, one can construct ν_i and thus V_{i+1} using $192(\log n+\log t)n$ queries, succeeding with probability at least $1-\frac{1}{nt}$. We start with $G_1 = G$ and iteratively apply Lemma 5 to construct the sequence $G_1,\ldots,G_t$ and $\nu_1,\ldots,\nu_t$. We apply Lemma 5 t times, and hence using a union bound, all of the G_i were successfully constructed with probability at least $1-t\frac{1}{nt} = 1-\frac{1}{n}$. Next, we show how to construct H^α assuming the sequences $G_1,\ldots,G_t$ and $\nu_1,\ldots,\nu_t$ are

valid. Note that constructing the graph decomposition we use at most $192(\log n + \log t)n \times t \in O(n \log^2 n + n \log^2 \max_{e\in E} \alpha w_e)$ queries.

Recall that $E_{\nu_i} = E_i \setminus E_{i+1}$. Also, we have $E_1 = E$. Thus, the sequence $E_{\nu_1}, \ldots, E_{\nu_{t-1}}$ is a decomposition of $E \setminus E_t$. Also, note that $E_{\nu_i} = \{(u,v) \big| u \in \nu_i \text{ and } v \in V_i\}$. Therefore, given $G_1, \ldots, G_t$ and $\nu_1, \ldots, \nu_t$ we can decompose the edge set E into $E_{\nu_1}, \ldots, E_{\nu_t}$.

Let j be the smallest index such that $L_j \leq \frac{1}{\alpha}$. Notice that $j \leq t$ by Inequality 4. For each $i \in \{1, \ldots, j-1\}$ we query each edge $e \in E_{\nu_i}$. If $\alpha w_e > 1$, add edge e with weight αw_e to E_H. If $\alpha w_e \leq 1$, we add edge e with weight 1 to E_H with probability αw_e independently.

For each $i \in \{j, \ldots, t-1\}$ we query each edge $e \in E_{\nu_i}$ with probability αL_i. We add a queried edge e to E_H with probability $\frac{w_e}{L_i}$ and withdraw it otherwise. Note that $\alpha L_i \leq \alpha L_j \leq \alpha \frac{1}{\alpha} = 1$. Also L_i is an upper bound on the edge weights in $E_i \supseteq E_{\nu_i}$, and thus $\frac{w_e}{L_i} \leq 1$. Therefore, the probabilities αL_i and $\frac{w_e}{L_i}$ are valid. Also, notice that we add each edge to E_H with probability $\alpha L_i \times \frac{w_e}{L_i} = \alpha w_e$ as desired.

For each edge $e \in E_t$ we query e with probability $\frac{2}{\lambda n}$. We add a queried edge e to E_H with probability $\frac{\lambda n}{2}\alpha w_e$ and withdraw it otherwise. Recall L_t is an upper bound on the edges edges weights in E_t, and by Inequality 4 we have $L_t \leq \frac{2}{\lambda \alpha n}$. Thus, we have $\frac{\lambda n}{2}\alpha w_e \leq \frac{\lambda n}{2}\alpha L_t \leq \frac{\lambda n}{2}\alpha \frac{2}{\lambda \alpha n} = 1$. Therefore, $\frac{\lambda n}{2}\alpha w_e$ is a valid probability. Again, notice that we add each edge to E_H with probability $\frac{2}{\lambda n} \times \frac{\lambda n}{2}\alpha w_e = \alpha w_e$ as desired. Next we bound the total number of edges that we query.

Let Y_e be a random variable that is 1 if we query e and 0 otherwise. We bound the expected number of edges that we query by

$$
\begin{aligned}
&\mathrm{E}\left[\sum_{e\in E} Y_e\right] = \sum_{e\in E} \mathrm{E}\left[Y_e\right] \\
&= \sum_{i=1}^{j-1} \sum_{e\in E_{\nu_i}} \mathrm{E}\left[Y_e\right] + \sum_{i=j}^{t-1} \sum_{e\in E_{\nu_i}} \mathrm{E}\left[Y_e\right] + \sum_{e\in E_t} \mathrm{E}\left[Y_e\right] \\
&= \sum_{i=1}^{j-1} \sum_{e\in E_{\nu_i}} 1 + \sum_{i=j}^{t-1} \sum_{e\in E_{\nu_i}} \alpha L_i + \sum_{e\in E_t} \frac{2}{\lambda n} \\
&\leq \sum_{i=1}^{t} \sum_{e\in E_{\nu_i}} \alpha L_i + \sum_{e\in E_t} \frac{2}{\lambda n} && \forall_{i<j} \alpha L_i \geq 1 \\
&\leq \sum_{i=1}^{t} \sum_{e\in E_{\nu_i}} \alpha L_i + \frac{n}{\lambda} && |E_t| \leq \binom{n}{2} \\
&\leq \alpha \sum_{i=1}^{t} |\nu_i||V_i| L_i + \frac{n}{\lambda} && |E_{\nu_i}| \leq |\nu_i||V_i| \\
&\leq \alpha \sum_{i=1}^{t} \sum_{e\in E_{\nu_i}} \frac{32}{\lambda} w_e + \frac{n}{\lambda} && \text{By Lemma 3} \\
&\leq \frac{32}{\lambda}\alpha \sum_{e\in E} w_e + \frac{n}{\lambda} && E_{\nu_1}, \ldots, E_{\nu_t} \subseteq E \text{ are disjoint} \\
&= \frac{32}{\lambda}\alpha \binom{n}{2}\overline{w} + \frac{n}{\lambda} \\
&\in O(\frac{1}{\lambda}\alpha \binom{n}{2}\overline{w} + \frac{n}{\lambda}).
\end{aligned}
$$

We used $O(n \log^2 n + n \log^2 \max_{e\in E} \alpha w_e)$ queries to construct the sequences $G_1, \ldots, G_t$ and $\nu_1, \ldots, \nu_t$, and used $O(\frac{1}{\lambda}\alpha\binom{n}{2}\overline{w} + \frac{n}{\lambda})$ queries to construct H^α based on these sequences. Therefore, in total we used $O\big(n \log^2 n + n \log^2 \max_{e\in E} \alpha w_e + \frac{1}{\lambda}\alpha\overline{w}\binom{n}{2} + \frac{n}{\lambda}\big)$ queries in expectation. □

C. Constructing H_β

The following lemma relates β with α. We use this to construct H_β using H^α.

Lemma 8: Let $\gamma \in [1, \infty)$ be an arbitrary number. Let $\frac{1}{\gamma}\overline{w} \leq \hat{w} \leq \overline{w}$, $\alpha = \frac{\beta}{\binom{n}{2}\hat{w}}$, and $H^\alpha = (V, E_H)$. We have

$$\beta \leq \mathrm{E}\left[\sum_{e\in E_H} w'_e\right] \leq \gamma\beta,$$

where w'_e is the weight of e in H^α.

Proof : We have

$$\mathrm{E}\left[\sum_{e\in E_H} w'_e\right] = \sum_{e\in E}\mathrm{E}\left[w'_e\right] = \sum_{e\in E}\alpha w_e$$
$$= \sum_{e\in E}\frac{\beta}{\binom{n}{2}\hat{w}}w_e = \frac{\beta}{\hat{w}}\frac{\sum_{e\in E}w_e}{\binom{n}{2}}$$
$$= \frac{\overline{w}}{\hat{w}}\beta.$$

This together with $\frac{1}{\gamma}\overline{w} \le \hat{w}$ gives us

$$\mathrm{E}\left[\sum_{e\in E_H} w'_e\right] = \frac{\overline{w}}{\hat{w}}\beta \le \gamma\beta.$$

Similarly, by applying $\hat{w} \le \overline{w}$ we have

$$\mathrm{E}\left[\sum_{e\in E_H} w'_e\right] = \frac{\overline{w}}{\hat{w}}\beta \ge \beta.$$

□

Lemma 9 shows how to estimate $\overline{w}$. We use this lemma together with Lemma 8 to find a proper α based on the desired β to construct H_β. We note that in a metric space, i.e. $\lambda = 1$, the following lemma gives a $1-\epsilon$ approximation of the average weight of the edges using $\tilde{O}(\frac{n}{\epsilon^2})$ queries, while the previous algorithm of Indyk [20] uses $O(\frac{n}{\epsilon^{3.5}})$ queries[3]. In the next section, using H_β we improve this lemma and estimate the average weight of the edges using only $\tilde{O}(n+\frac{1}{\epsilon^2})$ queries.

Lemma 9: For $\epsilon \in (0,1]$, one can find an estimator $\hat{w}$ of the average weight of the edges $\overline{w}$ such that $(1-\epsilon)\overline{w} \le \hat{w} \le (1+\epsilon)\overline{w}$, with probability $1-\frac{2}{n}$, using $O\big(n\log^2 n + \frac{n\log n}{\epsilon^2\lambda}\big) \in \tilde{O}(\frac{n}{\epsilon^2\lambda})$ queries.

Proof : We first use $O(\frac{n}{\lambda})$ queries to provide an estimate $\hat{w}'$ such that $\frac{1}{2n}\overline{w} \le \hat{w}' \le \overline{w}$. Next we set $\alpha = \frac{\beta}{\binom{n}{2}\hat{w}}$ and construct a corresponding H^α. We use Lemma 8 and Theorem 7 to lower bound the total weight of sampled edges by $\frac{3\log(2n)}{\epsilon^2}$ and upper bound the number of queries by $O\big(n\log^2 n + \frac{n\log n}{\epsilon^2\lambda}\big)$. At the end we use the lower bound on the total weight of sampled edges to show that the average weight of edges in H^α is concentrated around $\overline{w}$.

[3]Note that the algorithm in [20] works with a constant probability while our algorithm works with probability $1-\frac{1}{n}$. The previous algorithm requires an extra logarithmic factor to work with probability $1-\frac{1}{n}$.

Let v be an arbitrary vertex. We have

$$\sum_{u\in V\setminus\{v\}} w_{u,v}$$
$$= \frac{1}{n}\sum_{u\in V\setminus\{v\}} w_{u,v} + \frac{n-1}{n}\sum_{u\in V\setminus\{v\}} w_{u,v}$$
$$\ge \frac{1}{n}\sum_{u\in V\setminus\{v\}} w_{u,v} + \frac{n-1}{n}\frac{\lambda}{n-2}\sum_{u,u'\in V\setminus\{v\}} w_{u,u'}$$
$$> \frac{\lambda}{n}\sum_{e\in E} w_e$$

Hence for a set $S \subseteq V$ with $|S| = \lceil\frac{1}{\lambda}\rceil$ we have

$$\sum_{v\in S}\sum_{u\in V\setminus\{v\}} w_{u,v} > \sum_{v\in S}\frac{\lambda}{n}\sum_{e\in E} w_e \ge \frac{1}{n}\sum_{e\in E} w_e.$$

On the other hand every edge appears at most twice in $\sum_{v\in S}\sum_{u\in V\setminus\{v\}} w_{u,v}$ and hence we have $\sum_{v\in S}\sum_{u\in V\setminus\{v\}} w_{u,v} \le 2\sum_{e\in E} w_e$. Therefore, by setting $\hat{w}' = \frac{1}{2\binom{n}{2}}\sum_{v\in S}\sum_{u\in V\setminus\{v\}} w_{u,v}$ we have $\frac{1}{2n}\overline{w} \le \hat{w}' \le \overline{w}$. Hence, one can query at most $\frac{n}{\lceil\lambda\rceil}$ edges to find a number $\hat{w}'$ such that $\frac{1}{2n}\overline{w} \le \hat{w}' \le \overline{w}$.

Next, we set $\alpha = \frac{3\log(2n)}{\epsilon^2\binom{n}{2}\hat{w}'}$. By Lemma 8 we have

$$\frac{3\log(2n)}{\epsilon^2} \le \mathrm{E}\left[\sum_{e\in E_H} w'_e\right] \le \frac{6n\log(2n)}{\epsilon^2}, \qquad (5)$$

where w'_e is the weight of e in H^α. By Lemma 7, with probability $1-\frac{1}{n}$, the expected number of queries we need to construct H^α is at most

$$O\Big(n\log^2 n + n\log^2 \max_{e\in E}\alpha w_e + \frac{1}{\lambda}\alpha\overline{w}\binom{n}{2} + \frac{n}{\lambda}\Big) \in$$
$$O\Big(n\log^2 n + n\log^2\Big(\frac{n\log(n)}{\epsilon^2}\Big) + \frac{n\log(n)}{\epsilon^2\lambda} + \frac{n}{\lambda}\Big) \in$$
$$O\Big(n\log^2 n + \frac{n\log n}{\epsilon^2\lambda}\Big). \qquad \text{Assuming } \epsilon \ge \frac{1}{n} \text{ w.l.o.g.}$$

Now we set $\hat{w} = \frac{1}{\binom{n}{2}}\sum_{e\in E_H}\frac{w'_e}{\alpha}$, where w'_e is the weight of e in H^α. To complete the proof we show that $(1-\epsilon)\overline{w} \le \hat{w} \le (1+\epsilon)\overline{w}$, with probability $1-\frac{1}{n}$. Notice that

$$\mathrm{E}\left[\hat{w}\right] = \mathrm{E}\left[\frac{1}{\binom{n}{2}}\sum_{e\in E_H}\frac{w'_e}{\alpha}\right] = \frac{1}{\binom{n}{2}}\sum_{e\in E} w_e = \overline{w}. \qquad (6)$$

Let χ_e be a binary random variable that indicates

whether χ_e is sampled in H^α or not. Note that

$$\begin{aligned}\hat{w} - \mathrm{E}[\hat{w}] &= \frac{1}{\binom{n}{2}} \sum_{e\in E_H} \frac{w'_e}{\alpha} - \frac{1}{\binom{n}{2}} \sum_{e\in E} \mathrm{E}\left[\frac{w'_e}{\alpha}\right] \\ &= \frac{1}{\binom{n}{2}\alpha}\Big(\sum_{e\in E_H} w'_e - \sum_{e\in E} \mathrm{E}[w'_e]\Big) \\ &= \frac{1}{\binom{n}{2}\alpha}\Big(\sum_{w_e\le \frac{1}{\alpha}} \chi_e - \sum_{w_e\le \frac{1}{\alpha}} \mathrm{E}[w'_e]\Big), \quad (7)\end{aligned}$$

where the last equality is by the fact that $w'_e = \mathrm{E}[w'_e]$ when $w_e > \frac{1}{\alpha}$. We have

$$\begin{aligned}&\Pr[|\hat{w} - \overline{w}| \le \epsilon\overline{w}] \\ &= \Pr[|\hat{w} - \mathrm{E}[\hat{w}]\,| \le \epsilon\overline{w}] && \text{By Equality 6} \\ &= \Pr\left[\Big|\frac{1}{\binom{n}{2}\alpha}\Big(\sum_{w_e\le \frac{1}{\alpha}} \chi_e - \sum_{w_e\le \frac{1}{\alpha}} \mathrm{E}[w'_e]\Big)\Big| \le \epsilon\overline{w}\right] \\ & && \text{By Equality 7} \\ &= \Pr\left[\Big| \sum_{w_e\le \frac{1}{\alpha}} \chi_e - \sum_{w_e\le \frac{1}{\alpha}} \mathrm{E}[w'_e]\,\Big| \le \epsilon\alpha\binom{n}{2}\overline{w}\right] \\ &\le 2\exp\Big(-\frac{1}{3}\big(\frac{\epsilon\alpha\binom{n}{2}\overline{w}}{\sum_{w_e\le \frac{1}{\alpha}} \mathrm{E}[w'_e]}\big)^2 \sum_{w_e\le \frac{1}{\alpha}} \mathrm{E}[w'_e]\Big) \\ &\le 2\exp\Big(-\frac{1}{3}\frac{\epsilon^2\alpha^2\binom{n}{2}^2\overline{w}^2}{\sum_{w_e\le \frac{1}{\alpha}} \mathrm{E}[w'_e]}\Big) \\ &= 2\exp\Big(-\frac{1}{3}\frac{\epsilon^2\big(\sum_{e\in E} \mathrm{E}[w'_e]\big)^2}{\sum_{w_e\le \frac{1}{\alpha}} \mathrm{E}[w'_e]}\Big) \\ &\le 2\exp\Big(-\frac{1}{3}\epsilon^2 \sum_{e\in E} \mathrm{E}[w'_e]\Big) \\ &\le 2\exp\Big(-\frac{1}{3}\epsilon^2\frac{3\log(2n)}{\epsilon^2}\Big) && \text{By Inequality 5} \\ &= 2\exp\big(-\log(2n)\big) = \frac{1}{n}.\end{aligned}$$

This means that with probability $1 - \frac{1}{n}$ we have $(1 - \epsilon)\overline{w} \le \hat{w} \le (1 + \epsilon)\overline{w}$ as desired. □

The following theorem constructs H_β using $\tilde{O}(\frac{n+\beta}{\lambda})$ queries, with high probability.

Theorem 10: For any β one can construct H_β using expected $O(n\log^2 n + n\log^2\beta + \frac{\beta}{\lambda} + \frac{n\log n}{\lambda}) \in \tilde{O}(\frac{n+\beta}{\lambda})$ expected queries, with probability of success at least $1 - \frac{3}{n}$.

Proof : First, using Lemma 9 we find an estimator $\hat{w}$ of the average weight of the edges $\overline{w}$ such that $\frac{1}{2}\overline{w} \le \hat{w} \le \overline{w}$, with probability $1-\frac{2}{n}$, using $O(n\log^2 n + \frac{n\log n}{\lambda})$ expected queries. Lemma 8 says that by picking $\alpha = \frac{\beta}{\binom{n}{2}\hat{w}}$, we have $\beta \le \mathrm{E}\left[\sum_{e\in E_H} w'_e\right] \le 2\beta$, where w'_e is the weight of e in $H^\alpha = (V, E_H)$. [4] By Theorem 7 one can construct H^α using $O(n\log^2 n + n\log^2 \max_{e\in E} \alpha w_e + \frac{1}{\lambda}\alpha\overline{w}\binom{n}{2} + \frac{n}{\lambda})$ expected queries, with probability $1 - \frac{1}{n}$. Note that, we have

$$\begin{aligned}\max_{e\in E} \alpha w_e &= \alpha \max_{e\in E} w_e \\ &\le \alpha\binom{n}{2}\overline{w} && \overline{w} \ge \frac{\max_{e\in E} w_e}{\binom{n}{2}} \\ &= \frac{4\beta}{3\binom{n}{2}\hat{w}}\binom{n}{2}\overline{w} && \alpha = \frac{4\beta}{3\binom{n}{2}\hat{w}} \\ &\le 2\beta && \frac{2}{3}\overline{w} \le \hat{w}\end{aligned}$$

Also, we have

$$\begin{aligned}\alpha\overline{w}n^2 &= \frac{4\beta}{3\binom{n}{2}\hat{w}}\overline{w}\binom{n}{2} && \text{By } \alpha = \frac{4\beta}{3\binom{n}{2}\hat{w}} \\ &= \frac{4\overline{w}}{3\hat{w}}\beta \\ &\le 2\beta. && \text{By } \frac{2}{3}\overline{w} \le \hat{w}\end{aligned}$$

By $\alpha\overline{w}n^2 \le 2\beta$ and $\max_{e\in E} \alpha w_e \le 2\beta$ we have

$$\begin{aligned}&n\log^2 n + n\log^2 \max_{e\in E} \alpha w_e + \frac{1}{\lambda}\alpha\overline{w}\binom{n}{2} + \frac{n}{\lambda} \\ &\le n\log^2 n + nlog^2(2\beta) + \frac{2\beta}{\lambda} + \frac{n}{\lambda} \\ &\in O(n\log^2 n + n\log^2\beta + \frac{\beta+n}{\lambda}).\end{aligned}$$

Therefore, the total number of expected queries is $O(\frac{n\log n}{\lambda} + n\log^2 n + n\log^2\beta + \frac{\beta+n}{\lambda}) \in \tilde{O}(\frac{\beta+n}{\lambda})$. We properly estimate $\hat{w}$ with probability at least $1 - \frac{2}{n}$ and Theorem 7 holds with probability at least $1 - \frac{1}{n}$. Therefore, by the union bound, the statement of this theorem holds with probability at least $1 - \frac{3}{n}$. □

III. Applications of Linear Sampling

In this section we use the sketch H_β to develop approximation algorithms for densest subgraph, maximum k-hypermatching, and maximum cut, as well as estimating the average distance. We first define the problems and provide relevant notation. The densest subgraph of a graph $G = (V, E)$ is an induced subgraph of G, indicated by its set of vertices $S^* \subseteq V$, that maximizes $\frac{\sum_{u,v\in S^*} w_{u,v}}{|S^*|}$. We indicate the value of the densest subgraph by opt_D. The max cut of a graph

[4] Note that, for any $\eta \in (0, 1]$, one can use lemma 9 to find $\hat{w}$ such that $\frac{1}{1+\eta}\overline{w} \le \hat{w} \le \overline{w}$, with probability $1 - \frac{2}{n}$, using $O(n\log^2 n + \frac{n\log n}{\eta^2\lambda})$ expected queries, and then apply Lemma 8 to show that by picking $\alpha = \frac{\beta}{\binom{n}{2}\hat{w}}$, we have $\beta \le \mathrm{E}\left[\sum_{e\in E_H} w'_e\right] \le (1+\eta)\beta$. We use $\eta = 1$ throughout for convenience.

$G = (V, E)$ is a decomposition of the vertex set of G into two sets $S^*, V \setminus S^* \subseteq V$, that maximizes $\sum_{u \in S^*, v \in V \setminus S^*} w_{u,v}$. We indicate the value of the max cut by $\mathsf{opt_C}$. A k-hypermatching of a set of points V is a decomposition of V into a collection of n/k sets $\mathbb{S}^* = \{S_1^*, S_2^*, \ldots, S_{n/k}^*\}$, each of size k. One can also see this as covering a graph $G = (V, E)$ with clusters of size k. A maximum k-hypermatching is a k-hypermatching that maximizes $\sum_{i=1}^{n/k} \sum_{u,v \in S_i^*} w_{u,v}$. We use $\mathsf{opt_M}$ to indicate the value of the maximum k-hypermatching.

For a sketch $H_\beta = (V, E_H)$ we define random variables $X_{u,v}$ and $Y_{u,v}$. $Y_{u,v}$ is 0 if $(u, v) \notin E_H$, and is equal to the weight of the edge (u, v) in H_β otherwise. $X_{u,v} = 1$ if $Y_{u,v} = 1$ and $X_{u,v} = 0$ otherwise. Recall that if we sample an edge e with $\alpha w_e \leq 1$, weight of e in H_β is 1. Note that $\mathbf{E}[Y_{u,v}] = \alpha w_{u,v}$.

We first start with a simple application, using H_β to estimate the average weight of the edges using $\beta = O(\frac{\log n}{\varepsilon^2})$. This together with Theorem 10 allows us to find the average weight of the edges in a λ-metric space with probability $1 - \frac{4}{n}$ using $O(n \log^2 n + \frac{\log n}{\lambda \varepsilon^2} + \frac{n \log n}{\lambda}) \in \tilde{O}(\frac{n + 1/\varepsilon^2}{\lambda})$ expected queries.[5] In particular for a metric space this gives a $1 - \varepsilon$ approximation of the average weight of the edges using $\tilde{O}(n + \frac{1}{\epsilon^2})$ queries.

In what follows (throughout this section), when considering the failure probability of the approximation algorithms, we assume that H_β has been constructed successfully. That is, we provide for a failure probability in this stage of at most $1/n$, which when combined with Theorem 10 allows for our success probability of at least $1 - \frac{4}{n}$ overall. We present the proof of this theorem in the arxiv version [18].

Theorem 11: Take $\beta = \frac{3 \log(2n)}{\epsilon^2}$. We have

$$(1 - \varepsilon)\overline{w} \leq \frac{1}{\alpha \binom{n}{2}} \sum_{e \in E} Y_e \leq (1 + \varepsilon)\overline{w},$$

with probability at least $1 - \frac{1}{n}$.

Next we provide our results for the densest subgraph problem.

Theorem 12: Take $\beta = \frac{9 \log n}{\varepsilon^2} n$. Let S be a ϕ-approximation solution to the densest subgraph problem on H_β. S is a $\phi - 2\varepsilon$ approximation solution to the densest subgraph on G, with probability at least $1 - \frac{1}{n}$.

[5] Again, we emphasize that we can turn these results into bounds with a corresponding upper bound on the queries, with a small increase in the failure probability.

Proof : We start by lower bounding $\mathsf{opt_D}$.

$$\begin{aligned} \mathsf{opt_D} &\geq \frac{\sum_{u,v \in V} w_{u,v}}{|V|} = \frac{\frac{1}{\alpha} \sum_{u,v \in V} \mathbf{E}[Y_{u,v}]}{n} \\ &\geq \frac{1}{\alpha} \frac{\beta}{n} = \frac{1}{\alpha} \frac{9 \log n}{\varepsilon^2}. \end{aligned} \tag{8}$$

Let S' be a subset of V. We define $X_{S'} = \sum_{u,v \in S'} X_{u,v}$, and $Y_{S'} = \sum_{u,v \in S'} Y_{u,v}$. Note that we have $X_{S'} \leq Y_{S'}$. We have $\mathbf{E}[Y_{S'}] = \sum_{u,v \in S'} \mathbf{E}[Y_{u,v}] = \alpha \sum_{u,v \in S'} w_{u,v}$. Hence, we have

$$\mathsf{opt_D} \geq \frac{\sum_{u,v \in S'} w_{u,v}}{|S'|} = \frac{\mathbf{E}[Y_{S'}]}{\alpha |S'|} \geq \frac{\mathbf{E}[X_{S'}]}{\alpha |S'|} \tag{9}$$

Notice that $Y_e \neq X_e$ implies $Y_e = \mathbf{E}[Y_e]$ and hence $|Y_{S'} - \mathbf{E}[Y_{S'}]| = |X_{S'} - \mathbf{E}[X_{S'}]|$. Also, note that $X_{u,v}$'s are chosen independently, and hence by applying the Chernoff bound to $X_{S'}$ for $\epsilon = \varepsilon \frac{\alpha \mathsf{opt_D} |S'|}{\mathbf{E}[X_{S'}]}$ we have

$$\begin{aligned} &\Pr\left[|Y_{S'} - \mathbf{E}[Y_{S'}]| \geq \varepsilon \alpha \mathsf{opt_D} |S'|\right] \\ &= \Pr\left[|X_{S'} - \mathbf{E}[X_{S'}]| \geq \varepsilon \alpha \mathsf{opt_D} |S'|\right] \\ &\leq 2 \exp\Big(- \frac{1}{3} \big(\varepsilon \frac{\alpha \mathsf{opt_D} |S'|}{\mathbf{E}[X_{S'}]}\big)^2 \mathbf{E}[X_{S'}] \Big) && \text{Chernoff} \\ &= 2 \exp\Big(- \frac{1}{3} \varepsilon^2 \frac{\alpha^2 \mathsf{opt_D}^2 |S'|^2}{\mathbf{E}[X_{S'}]} \Big) \\ &\leq 2 \exp\Big(- \frac{1}{3} \varepsilon^2 \alpha \mathsf{opt_D} |S'| \Big) && \text{Ineq. 9} \\ &\leq 2 \exp\Big(- \frac{1}{3} \varepsilon^2 \frac{9 \log n}{\varepsilon^2} |S'| \Big) && \text{Ineq. 8} \\ &= 2 \exp\big(- 3 |S'| \log n \big). \end{aligned}$$

Next we union bound over all choices of S'.

$$\begin{aligned} &\Pr\left[\exists_{S'} \big| Y_{S'} - \mathbf{E}[Y_{S'}] \big| \geq \varepsilon \alpha \mathsf{opt_D} |S'|\right] \\ &= \Pr\left[\exists_k \exists_{|S'|=k} \big| Y_{S'} - \mathbf{E}[Y_{S'}] \big| \geq \varepsilon \alpha \mathsf{opt_D} k\right] \\ &\leq \sum_{k=2}^{n} \Pr\left[\exists_{|S'|=k} \big| Y_{S'} - \mathbf{E}[Y_{S'}] \big| \geq \varepsilon \alpha \mathsf{opt_D} k\right] \\ &\leq \sum_{k=2}^{n} \sum_{|S'|=k} \Pr\left[\big| Y_{S'} - \mathbf{E}[Y_{S'}] \big| \geq \varepsilon \alpha \mathsf{opt_D} k\right] \\ &\leq \sum_{k=2}^{n} \sum_{|S'|=k} 2 \exp\big(- 3k \log n \big) \\ &= \sum_{k=2}^{n} 2 \binom{n}{k} \exp\big(- 3k \log n \big) \\ &\leq \sum_{k=2}^{n} 2 \exp\big(- 3k \log n + k \log n \big) && \binom{n}{k} \leq n^k \end{aligned}$$

$$\leq \sum_{k=2}^{n} 2\exp\big(-4\log n\big) \qquad k \geq 2$$
$$\leq 2\exp\big(-3\log n\big)$$
$$= \frac{2}{n^3} < \frac{1}{n}. \qquad n \geq 2$$

Therefore, with probability at least $1-\frac{1}{n}$ simultaneously for all $S' \subseteq V$ we have

$$\big|Y_{S'} - \mathbf{E}[Y_{S'}]\big| \leq \varepsilon\alpha\mathsf{opt}_{\mathsf{D}}|S'|. \tag{10}$$

Next we prove the statement of the theorem in the cases where Inequality 10 holds. Let S^* be a densest subgraph of G. We have

$$\frac{\sum_{u,v\in S} w_{u,v}}{|S|}$$
$$= \frac{\frac{1}{\alpha}\mathbf{E}[\sum_{u,v\in S} Y_{u,v}]}{|S|} \qquad \mathbf{E}[Y_{u,v}] = \alpha w_{u,v}$$
$$= \frac{1}{\alpha}\frac{\mathbf{E}[Y_S]}{|S|} \qquad \text{Definition of } Y_S$$
$$\geq \frac{1}{\alpha}\frac{Y_S}{|S|} - \varepsilon\mathsf{opt}_{\mathsf{D}} \qquad \text{By Inequality 10}$$
$$\geq \frac{1}{\alpha}\phi\max_{S''}\frac{Y_{S''}}{|S''|} - \varepsilon\mathsf{opt}_{\mathsf{D}} \qquad S \text{ is } \phi \text{ approx. on } H_\beta$$
$$\geq \frac{1}{\alpha}\phi\frac{Y_{S^*}}{|S^*|} - \varepsilon\mathsf{opt}_{\mathsf{D}}$$
$$\geq \frac{1}{\alpha}\phi\frac{\mathbf{E}[Y_{S^*}]}{|S^*|} - 2\varepsilon\mathsf{opt}_{\mathsf{D}} \qquad \text{By Inequality 10}$$
$$\geq \phi\frac{\sum_{u,v\in S^*} w_{u,v}}{|S^*|} - 2\varepsilon\mathsf{opt}_{\mathsf{D}} \qquad \mathbf{E}[Y_{u,v}] = \alpha w_{u,v}$$
$$= (\phi - 2\varepsilon)\mathsf{opt}_{\mathsf{D}}. \qquad \text{Definition of } S^*$$

□

Recall that, as stated in the introduction, this result implies a $(1/2-\epsilon)$-approximation algorithm for densest subgraph in λ-metric spaces requiring $\tilde{O}(\frac{n}{\lambda\epsilon^2})$ time.

The following theorem shows the efficiency of our technique for k-hypermatching. The proof of this theorem is presented in the arxiv version [18].

Theorem 13: Choose $\beta = \frac{6\log n}{\varepsilon^2}\frac{n^2}{k-1} \in \tilde{O}\big(\frac{n^2}{\epsilon^2 k}\big)$. Let $\mathbb{S} = \{S_1, S_2, \ldots, S_{n/k}\}$ be a ϕ-approximation solution to the k-hypermatching on unweighted graph H_β. $\mathbb{S}$ is a $\phi - 2\varepsilon$ approximation solution to the k-hypermatching on G, with probability at least $1-\frac{1}{n}$.

Finally we show the efficiency of our sketch for finding the maximum cut, again following the same basic proof outline. Here, we indicate a cut by the set of vertices of its smaller side, breaking ties arbitrarily. The proof of this theorem is presented in the arxiv version [18].

Theorem 14: Choose $\beta = \frac{18\log n}{\varepsilon^2}n$. Let S be a ϕ-approximation solution to the maximum cut on H_β. S is a $\phi - 2\varepsilon$ approximation solution to the maximum cut on G, with probability at least $1-\frac{1}{n}$.

IV. Impossibility Results

In this section we consider all of the problems of the previous section and show that it is necessary to use $\Omega(n)$ queries even if we just want to estimate the value of the solutions. In particular, we show that $\Omega(n)$ queries are required to distinguish the following two graphs.

- In G_1 we have n vertices $\{v_1, \ldots, v_n\}$ and the weight of all edges are 0.
- In G_2 again we have n vertices. Pick an index $r \in \{1, \ldots, n\}$ uniformly at random. The weight of each edge neighboring v_r is 1. The weight of all other edges is 0.

The following lemma shows the hardness of distinguishing G_1 and G_2.

Lemma 15: For any $\delta \in (0, 0.5]$, it is impossible to distinguish G_1 and G_2 using $\delta n - 1$ queries with probability $0.5 + \delta$.

Proof : Let Alg be a (possibly randomized) algorithm that distinguishes G_1 and G_2 using at most $\delta n - 1$ queries. For simplicity, and without loss of generality, we assume that Alg makes exactly $\delta n - 1$ queries. Let $(u_1, u_2), (u_3, u_4), \ldots, (u_{k-1}, u_k)$ be the sequence of edges probed by Alg, where the u_i's may be random variables and $k = 2\delta n - 2$. Notice that v_r is chosen uniformly at random. Hence, in case that the input is G_2, for any arbitrary $j \in \{1, \ldots, k\}$ we have $\Pr[u_j = v_r] = \frac{1}{n}$. Therefore, we have

$$\Pr\Big[\exists_{i\in\{1,\ldots,\frac{k}{2}\}} w_{u_{2i-1},u_{2i}} \neq 1\Big]$$
$$= \Pr\big[\exists_{j\in\{1,\ldots,k\}} u_j = v_r\big]$$
$$\leq \sum_{j=1}^{k}\Pr[u_i = v_r] \qquad \text{By union bound}$$
$$= \frac{k}{n} \qquad \text{Since } \Pr[u_j = v_r] = \frac{1}{n}$$
$$< 2\delta. \qquad \text{Since } k = 2\delta n - 2$$

Hence, in the case that the input is G_2, with probability at least $1-2\delta$ all the edges that Alg queries have weight 0. Trivially, in the case that the input is G_1 all the queried edges have weight 0. Therefore the probability that Alg distinguishes G_1 and G_2 is less than $2\delta + \frac{1-2\delta}{2} = 0.5+\delta$. □

Lemma 15 implies the following theorem. We refer to the arxiv version [18] for details.

Theorem 16: Any approximation algorithm that estimates the solution of any of following problems in a metric graphs within any multiplicative factor with probability 0.51 requires $\Omega(n)$ queries.

- average distance
- densest subgraph
- maximum matching
- max cut

V. Conclusion

We have shown that in metric graphs one can efficiently obtain a linear sampling with a sublinear number of edge queries, allowing efficient sparsification that leads to efficient approximation algorithms. We believe this technique may be useful in generating approximation algorithms for other problems beyond those considered here. Open questions include possibly improving the lower bounds, and bridging the gap between the upper and lower bounds on required queries.

References

[1] N. Alon, S. Dar, M. Parnas, and D. Ron. Testing of clustering. *SIAM Journal on Discrete Mathematics*, 16(3):393–417, 2003.

[2] T. Andreae and H.J. Bandelt. Performance guarantees for approximation algorithms depending on parametrized triangle inequalities. *SIAM Journal on Discrete Mathematics*, 8(1):1–16, 1995.

[3] S. Assadi. Tight space-approximation tradeoff for the multi-pass streaming set cover problem. In *Symposium on Principles of Database Systems*, pp. 321–335, 2017.

[4] M. Bădoiu, A. Czumaj, P. Indyk, and C. Sohler. Facility location in sublinear time. In *International Colloquium on Automata, Languages, and Programming*, pp. 866–877, 2005.

[5] M. Bateni, H. Esfandiari, and V. Mirrokni. Almost optimal streaming algorithms for coverage problems. *arXiv preprint arXiv:1610.08096*, 2016.

[6] M. Bateni, H. Esfandiari, and V. Mirrokni. Distributed coverage maximization via sketching. *arXiv preprint arXiv:1612.02327*, 2016.

[7] A. Bhaskara, S. Daruki, and S. Venkatasubramanian. Sublinear algorithms for maxcut and correlation clustering. *arXiv preprint arXiv:1802.06992*, 2018.

[8] A. Bhaskara, M. Ghadiri, V. Mirrokni, and O. Svensson. Linear relaxations for finding diverse elements in metric spaces. In *Advances in Neural Information Processing Systems*, pp. 4098–4106, 2016.

[9] S. Bhattacharya, M. Henzinger, D. Nanongkai, and C. Tsourakakis. Space-and time-efficient algorithm for maintaining dense subgraphs on one-pass dynamic streams. In *Symposium on Theory of Computing*, pp. 173–182, 2015.

[10] M. Charikar. Greedy approximation algorithms for finding dense components in a graph. In *International Workshop on Approximation Algorithms for Combinatorial Optimization*, pp. 84–95, 2000.

[11] R. Chitnis, G. Cormode, H. Esfandiari, M. Hajiaghayi, A. McGregor, M. Monemizadeh, and S. Vorotnikova. Kernelization via sampling with applications to finding matchings and related problems in dynamic graph streams. In *Symposium on Discrete Algorithms*, pp. 1326–1344, 2016.

[12] R. Chitnis, G. Cormode, M. Hajiaghayi, and M. Monemizadeh. Parameterized streaming: Maximal matching and vertex cover. In *Symposium on Discrete Algorithms*, pp. 1234–1251, 2015.

[13] R. Cox, F. Dabek, F. Kaashoek, J. Li, and R. Morris. Practical, distributed network coordinates. *ACM SIGCOMM Computer Communication Review*, 34(1):113–118, 2004.

[14] A. Czumaj and C. Sohler. Sublinear-time approximation for clustering via random sampling. In *International Colloquium on Automata, Languages, and Programming*, pp. 396–407, 2004.

[15] A. Czumaj and C. Sohler. Small space representations for metric min-sum k-clustering and their applications. In *Symposium on Theoretical Aspects of Computer Science*, pp. 536–548, 2007.

[16] A. Czumaj and C. Sohler. Sublinear-time algorithms. In *Property Testing*, pp. 41–64. 2010.

[17] H. Esfandiari, M. Hajiaghayi, and D.P. Woodruff. Applications of uniform sampling: Densest subgraph and beyond. *arXiv preprint arXiv:1506.04505*, 2015.

[18] H. Esfandiari and M. Mitzenmacher. Metric Sublinear Algorithms via Linear Sampling. *arXiv preprint arXiv:1807.09302*, 2018.

[19] S. Har-Peled, P. Indyk, S. Mahabadi, and A. Vakilian. Towards tight bounds for the streaming set cover problem. In *Symposium on Principles of Database Systems*, pp. 371–383, 2016.

[20] P. Indyk. Sublinear time algorithms for metric space problems. In *Symposium on Theory of Computing*, pp. 428–434, 1999.

[21] P. Indyk. A sublinear time approximation scheme for clustering in metric spaces. In *Symposium on Foundations of Computer Science*, pp. 154–159, 1999.

[22] M. Kapralov, S. Khanna, and M. Sudan. Streaming lower bounds for approximating max-cut. In *Symposium on Discrete Algorithms*, pp. 1263–1282, 2015.

[23] M. Kapralov, S. Khanna, M. Sudan, and A. Velingker. (1+ ω (1))-approximation to max-cut requires linear space. In *Symposium on Discrete Algorithms*, pp. 1703–1722, 2017.

[24] D. Kogan and R. Krauthgamer. Sketching cuts in graphs and hypergraphs. In *Innovations in Theoretical Computer Science*, pp. 367–376, 2015.

[25] A. McGregor, D. Tench, S. Vorotnikova, and H. Vu. Densest subgraph in dynamic graph streams. In *International Symposium on Mathematical Foundations of Computer Science*, pp. 472–482, 2015.

[26] M. Mitzenmacher, J. Pachocki, R. Peng, C. Tsourakakis, and S. Xu. Scalable large near-clique detection in large-scale networks via sampling. In *International Conference on Knowledge Discovery and Data Mining*, pp. 815–824, 2015.

[27] M. Mitzenmacher and E. Upfal. *Probability and computing: Randomized algorithms and probabilistic analysis*. Cambridge University Press, 2005.

[28] M. Monemizadeh and D. Woodruff. 1-pass relative-error l_p-sampling with applications. In *Symposium on Discrete Algorithms*, pp. 1143–1160, 2010.

[29] Shanmugavelayutham Muthukrishnan. *Data streams: Algorithms and applications*. Now Publishers Inc, 2005.

[30] P. Pietzuch, J. Ledlie, M. Mitzenmacher, and M. Seltzer. Network-aware overlays with network coordinates. In *International Workshop on Dynamic Distributed Systems*, pp. 12–12, 2006.

[31] Y. Shavitt and T. Tankel. Big-bang simulation for embedding network distances in Euclidean space. *IEEE/ACM Transactions on Networking*, 12(6):993–1006, 2004.

[32] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei. Line: Large-scale information network embedding. In *International Conference on the World Wide Web*, pp. 1067–1077, 2015.

[33] S. Zadeh, M. Ghadiri, V.S. Mirrokni, and M. Zadimoghaddam. Scalable feature selection via distributed diversity maximization. In *AAAI Conference on Artificial Intelligence*, pp. 2876–2883, 2017.