Spectral sparsification via random spanners

[Extended Abstract]

Michael Kapralov*
Stanford iCME
Stanford, CA, USA
kapralov@stanford.edu

Rina Panigrahy
MSR Silicon Valley
Mountain View, CA, USA
rina@microsoft.com

ABSTRACT

In this paper we introduce a new notion of distance between nodes in a graph that we refer to as *robust connectivity*. Robust connectivity between a pair of nodes u and v is parameterized by a threshold κ and intuitively captures the number of paths between u and v of length at most κ. Using this new notion of distances, we show that any black box algorithm for constructing a spanner can be used to construct a spectral sparsifier. We show that given an undirected weighted graph G, simply taking the union of spanners of a few (polylogarithmically many) random subgraphs of G obtained by sampling edges at different probabilities, after appropriate weighting, yields a spectral sparsifier of G. We show how this be done in $\tilde{O}(m)$ time, producing a sparsifier with $\tilde{O}(n/\epsilon^2)$ edges. While the cut sparsifiers of Benczur and Karger are based on weighting edges according to (inverse) strong connectivity, and the spectral sparsifiers are based on resistance, our method weights edges using the robust connectivity measure. The main property that we use is that this new measure is always greater than the resistance when scaled by a factor of $O(\kappa)$ (κ is chosen to be $O(\log n)$), but, just like resistance and connectivity, has a bounded sum, i.e. $\tilde{O}(n)$, over all the edges of the graph.

1. INTRODUCTION

Large scale graphs are now a widely used tool for representing real world data. Many modern applications, such as search engines or social networks, require supporting various queries on large-scale graphs efficiently. An important primitive is maintaining a succint representation that preserves certain properties of the graph. In particular, one may be interested in supporting queries of some notion of distance between nodes in the graph. Various notions of distance between nodes of the graph have been considered recently, e.g. shortest path distance, minimum cuts, effective resistance etc. For all of these notions of 'distance' it is known how to compress a graph to a small representation that allows to compute 'distance' queries approximately from the compressed representation. For example, for shortest path distance this is provided by the spanner construction algorithms of Thorup and Zwick[14] (see also [15, 11, 2, 3, 4, 5]), for cuts this is given by sparsifiers of Benczúr and Karger[7] (see also [9]) and for effective resistances by spectral sparsifiers of Spielman and Srivastava [13] (see also [6]). In fact, all of these methods for succint representation satisfy a stronger guarantee – they support efficient queries of the corresponding 'distances' between nodes. It should be noted here that the latter two measures of distance essentially take into account *the set of paths* between a pair of nodes in the graph as opposed to the path of minimum length.

In this paper we introduce a new notion of distance between nodes in a graph that we refer to as *robust connectivity*. Robust connectivity between a pair of nodes u and v is parameterized by a threshold κ and intuitively captures the number of paths between u and v of length at most κ. Using this new notion of distances, we show that any black box algorithm to construct a spanner can be viewed as an algorithm to construct a sparsifier: Given a graph G, simply take the spanners of a few (polylogarithmically many) random subgraphs of G obtained by sampling edges at different probabilities; the union of these spanners, appropriately weighted, is also a sparsifier of G. While the cut sparsifiers of Benczur and Karger were based on weighting edges according to (inverse) strong connectivity, and the spectral sparsifiers were based on resistance, our method weights edges using the robust connectivity measure. The main property that we use is that this new measure is always greater than the resistance when scaled by a factor of $O(\kappa)$, but (just like resistance and connectivity) – has a bounded sum ($\tilde{O}(n)$) over all the edges of the graph. Our distance measure can be viewed as a generlization of the affinity measure that was used in an experimental paper to study user behavior in social networks [12].

We now give an outline of our results. In order to simplify presentation, we now define robust connectivities for unweighted graphs. The definition for weighted graphs will be given in Section 3. We stress here that even though we give the definition for unweighted graphs now, we will prove all our results for weighted graphs. Let $G = (V, E)$ be an undirected graph. For a sampling probability $p \in (0, 1)$ denote by G_p the graph obtained by sampling edges of G with probability p. Let $d_G(u, v)$ denote the shortest path distance between u and v in G. For a pair of nodes, the *κ-Robust Connectivity* is the highest sampling probability at which the they are at least distance κ apart in G_p with constant probability.

DEFINITION 1 (ROBUST CONNECTIVITY). *For a pair of nodes (u, v) let the κ-robust connectivity $q_\kappa(u, v)$ denote the largest $\eta \in (0, 1]$ such that $Pr[d_{G_\eta}(u, v) \geq \kappa] \geq 1/2$. For an edge $e = (u, v)$, we use $q_\kappa(e)$ to denote $q_\kappa(u, v)$.*

We show that the robust connectivity upper bounds the resistance

*Research supported in part by NSF award IIS-0904325. Part of the work was done when the author was an summer intern at Microsoft Research Silicon Valley.

ITCS '12 January 8-10 Cambridge, Massachusetts USA

(upto a $O(\kappa)$ factor) and also has a bounded sum over all edges.

LEMMA 2. *For all $e \in E$*

$$q_\kappa(e) \geq R_e/(2\kappa).$$

and

$$\sum_{e \in E} q_\kappa(e) \leq 2n^{1+O(1/\kappa)}$$

Further, based on distance oracles, which support approximate shortest path distance queries efficiently, one can construct an oracle that supports queries of approximate - robust connectivities between any pair of nodes.

LEMMA 3. *For a fixed graph G, there is a oracle that for any pair of nodes (u, v), can be used to query an estimate $\hat{q}_\kappa(u, v)$ of κ-robust connectivity. The estimate satisfies the (slightly weaker) conditions: for all $e \in E$*

$$\hat{q}_\kappa(e) \geq R_e/(2\kappa^2).$$

and

$$\sum_{e \in E} \hat{q}_\kappa(e) \leq O(n^{1+O(1/\kappa)})$$

If $\kappa = \Omega(\log n)$, the oracle can be constructed in time $\tilde{O}(m)$, stores a sketch of size $\tilde{O}(1)$ per node, and each query takes $\tilde{O}(1)$ time.

We note that Lemma 3 immediately yields a simple $\tilde{O}(m)$ time algorithm for spectral sparsification.

Let S be any black box algorithm spanner construction algorithm, that on input G outputs a spanner $S(G)$. Assume that the distances between all pairs of nodes in $S(G)$ are within factor $O(\log n)$ of the true distance in G. Existing spanner construction algorithms produce such spanners with $O(n)$ edges in time $\tilde{O}(m)$. We show

THEOREM 4. *Let $\{G_{i,j} : 1 \leq i \leq O\left(\log_{\frac{1}{1-\epsilon}}\left(\frac{n^4 w_{max}}{w_{min}}\right)\right)$, $1 \leq j \leq O(\log^3 n/\epsilon^3)\}$ be a collection of random subgraphs of G, where $G_{i,j}$ is an independent copy of G_p for $p = \frac{1}{w_{min}}(1-\epsilon)^i$. Then there is a weighting of the edges of the subgraph $H = \cup_{i,j} S(G_{i,j})$ such that it is a $(1\pm\epsilon)$-sparsifier of G. Moreover, such a weighting can be constructed in $\tilde{O}(m)$ time.*

Organization.

We start by introducing definitions related to spectral sparsification and spanners in section 2. In section 3 we give the definition of robust connectivities and prove Lemma 2 and Lemma 3, thus obtaining a simple algorithm for spectral sparsification. Finally, in section 4 we prove Theorem 4.

2. BACKGROUND AND NOTATION

For a weighted undirected graph $G = (V, E, w)$ the Laplacian matrix of G is the matrix defined as

$$L_G(i, j) = -w_{(i,j)} \text{ and } L_G(i, i) = \sum_{j \neq i} w_{ij}$$

DEFINITION 5 (SPECTRAL ORDERING OF GRAPHS). *We define a partial ordering $\prec$ on graphs by letting*

$$G \prec H \text{ if and only if } x^T L_G x \leq x^T L_H x \; \forall x \in \mathbb{R}^{|V|}.$$

A weighted undirected graph G can be associated with an electrical network with link e having conductance w_e (i.e. corresponding to a resistor of resistance $1/w_e$). Then the effective resistance R_e across an edge e is the potential difference induced across it when a unit current is injected at one end of e and extracted at the other end of e. We will use the following

THEOREM 6 (SPECTRAL SPARSIFICATION, [13]). *Let H be obtained by sampling edges of G independently with probability $p_e = \Theta(w_e R_e \log n/\epsilon^2)$ for some $\epsilon > 1/\sqrt{n}$ and giving each sampled edge weight $1/p_e$. Then whp*

$$(1-\epsilon)G \prec H \prec (1+\epsilon)G.$$

The following corollary is well-known (see, e.g.[10]):

COROLLARY 7 (OVERSAMPLING). *Let H be obtained by sampling edges of G independently with probability $p_e \geq c w_e R_e \log n/\epsilon^2$ for some $\epsilon > 1/\sqrt{n}$ and a sufficiently large constant $c > 0$, and giving each sampled edge weight $1/p_e$. Then whp*

$$(1-\epsilon)G \prec H \prec (1+\epsilon)G.$$

We will also need definitions related to spanners.

DEFINITION 8 (t-SPANNER). *A t-spanner of a weighted undirected graph G is a subgraph H of G such that the distances in H are stretch t estimates of the distances in G, i.e. for all $u, v \in V$ one has*

$$d_G(u, v) \leq d_H(u, v) \leq t \cdot d_G(u, v).$$

3. ROBUST CONNECTIVITIES

In this section we define robust connectivities, prove their main properties and give an efficient algorithm for approximating them. These results together with Corollary 7 imply a simple algorithm for spectral sparsification.

Let $G = (V, E)$ be a weighted undirected graph with edge weights w_e. For a sampling parameter $p \in \mathbb{R}^+$ denote by G_p the *unweighted* random graph obtained by sampling each edge $e \in E$ independently with probability $\min\{w_e p, 1\}$. For a graph H we denote the shortest path distance between nodes u and v in H by $d_H(u, v)$. We start by defining *robust connectivities*, for which we need an auxiliary definition.

DEFINITION 9. *Let $G = (V, E)$ be a weighted undirected graph. For $\eta \in \mathbb{R}^+$ and $e \in E$ let $p_\kappa(e, \eta) = \mathbf{Pr}[d_{G_\eta}(u, v) > \kappa]$.*

Note that when G is an unweighted graph, it is sufficient to consider $\eta \in (0, 1]$.

DEFINITION 10. *For $e = (u, v) \in E$ let the κ-robust connectivity $q_\kappa(e)$ denote the largest $\eta \in \mathbb{R}$ such that $p_\kappa(e, \eta) \geq 1/2$.*

In what follows the parameter κ will be fixed, and we will use the term robust connectivity for clarity. We will now show that $q_\kappa(e)$ upper bounds effective resistance up to a factor of κ. We will need

LEMMA 11 (RAYLEIGH'S MONOTONICITY PRINCIPLE, [8]). *Cutting an edge of an electrical resistor network does not decrease the effective resistance between any pair of nodes.*

LEMMA 12. *For all $e \in E$*

$$q_\kappa(e) \geq R_e/(2\kappa).$$

PROOF. We use the characterization of conductance between a pair of nodes in an electrical resistor network as

$$C_{uv} = \min_{x \in \mathbb{R}^V : x_s = 1, x_t = 0} \sum_{e=(u,v) \in E} w_e (x_u - x_v)^2. \quad (1)$$

For a sampling parameter $p \in \mathbb{R}^+$ let E_p denote a random set of edges obtained by sampling E independently with probability $\min\{w_e p, 1\}$. Similarly, denote the conductance of $e = (u, v)$ in G_p by

$$C_{uv}(p) = \min_{x \in \mathbb{R}^V : x_s = 1, x_t = 0} \sum_{e=(u,v) \in E_p} (x_u - x_v)^2. \quad (2)$$

We have

$$\begin{aligned} \mathbf{E}\left[C_{uv}(p)\right] &= \mathbf{E}\left[\min_{x \in \mathbb{R}^V : x_s = 1, x_t = 0} \sum_{e=(u,v) \in E_p} (x_u - x_v)^2\right] \\ &\leq \min_{x \in \mathbb{R}^V : x_s = 1, x_t = 0} \mathbf{E}\left[\sum_{e=(u,v) \in E_p} (x_u - x_v)^2\right] \\ &= \min_{x \in \mathbb{R}^V : x_s = 1, x_t = 0} \sum_{e=(u,v) \in E} (w_e p)(x_u - x_v)^2 = p C_{uv} \end{aligned}$$

and hence by Markov's inequality, one has

$$\mathbf{Pr}[C_{uv}(p) > 2p/R_e] \leq \mathbf{Pr}[C_{uv}(p) > 2\mathbf{E}[C_{uv}]] \leq 1/2. \quad (3)$$

On the other hand, for $p \geq q_\kappa(e)$ one has $\mathbf{Pr}[d_{G_p}(u, v) \leq \kappa] \geq 1/2$. This implies by Lemma 11 that

$$\mathbf{Pr}[C_{uv}(p) \geq 1/\kappa] \geq 1/2 \quad (4)$$

since the conductance of a path of length at most κ is at least $1/\kappa$. Setting $p = q_\kappa(e)$ and combining (3) and (4), we get that

$$2q_\kappa(e)/R_e \geq 1/\kappa, \quad (5)$$

i.e. $q_\kappa(e) \geq R_e/(2\kappa)$. □

We will need the following simple lemma

LEMMA 13. *For all $r' < r$ one has $p_\kappa(e, r) \leq p_\kappa(e, r')$.*

PROOF. The result follows by coupling the process of sampling edges at rate r' and r such that the set of edges picked by the first process is a subset of edges picked by the second process. □

We now prove that the sum of distance thresholds over edges of G is small. In order to do that, we relate the distance threshold of $e \in E$ to the probability of including e in a randomized spanner construction which we now define. Note that this construction is only used to prove an upper bound on the sum of distance thresholds and, in particular, we do not provide an efficient implementation here.

The intuition behind the randomized construction of the spanner is very simple, and we outline it here for the case of unweighted graphs. Arrange edges of G in a random order, add for each edge e in that order add e to the spanner H if e does not form a cycle of length smaller than κ with the edges that are included so far. It is known that such a process produces a spanner [1] with at most $n^{1+O(1/\kappa)}$ edges. On the other hand, the set of edges that have rank at most t in a uniformly random order essentially form a random sample of edges of G where each edge is present with probability t/m. Hence, the probability that the endpoints of e are at least κ apart in the subgraph formed by edges with rank at most t, and hence in the spanner constructed so far, is approximately $p_\kappa(e, t/m)$, allowing us to conclude that each edge is taken with probability about $q_\kappa(e)$, since it is taken with constant probability if it arrives before time $q_\kappa(e) \cdot m$. We now make a version of this argument formal. Since ordering edges by a uniformly random permutation causes dependencies, we choose a slightly different but essentially equivalent approach. We first define the following

Algorithm 1: Randomized spanner construction

```
H ← (V, ∅)
for η = 0 to 1/w_min with step Δ do
    for each e = (u, v) ∈ E do
        γ ← min{1, w_e Δ / (1 − w_e η)}.
        X_e(η) ← Bernoulli(γ).
        if X_e(η) = 1 then
            Add e to H if d_H(u, v) < κ.
        end if
    end for
end for
```

Note that intuitively, the sampling parameter η iterates over the relevant range of sampling parameters $[0, 1/w_{min}]$ with a sufficiently small step Δ, sampling the edges of G in such a way that at each time η the distribution of sampled edges is the same as in G_η. For sufficiently small Δ, we have that $\sum_e X_e(\eta) \leq 1$ for all η with high probability. More precisely, we say that an edge $e \in E$ is *sampled by time η* by Algorithm 1 if $X_e(\eta') = 1$ for at least one $\eta' \in [0, \eta]$. Note that an edge e that is sampled is not necessarily included in H. Denote the set of edges sampled by Algorithm 1 by time η by $E_s(\eta)$. Denote the set of edges included in H by time η by $E_H(\eta)$. The crucial property of Algorithm 1 that we will use in our analysis is

LEMMA 14. *$E_s(\eta)$ is distributed as a uniformly random independent sample of edges of E, where edge $e \in E$ is picked with probability $\min\{1, w_e\eta\}$.*

PROOF. Since the coin tosses for different edges are independent, it is sufficient to consider a fixed $e \in E$. We prove by induction on η (recall that η iterates over a discrete set) that

$$\mathbf{Pr}[e \in E_s(\eta)] = \min\{1, w_e\eta\}.$$

Base:$\eta = 0$ One has $\mathbf{Pr}[e \in E_s(\eta)] = \min\{1, w_e\eta\} = 0$.

Inductive step:$\eta \to \eta + \Delta$ One has

$$\begin{aligned} \mathbf{Pr}[e \in E_s(\eta + \Delta)] &= \mathbf{Pr}[e \in E_s(\eta)] \\ &+ \mathbf{Pr}[e \text{ sampled at step } \eta + \Delta | e \notin E_s(\eta)] \mathbf{Pr}[e \notin E_s(\eta)] \\ &= \mathbf{Pr}[e \in E_s(\eta)] \\ &+ \mathbf{Pr}[e \text{ sampled at step } \eta + \Delta] \mathbf{Pr}[e \notin E_s(\eta)] \\ &= w_e\eta + \min\left\{1, \frac{w_e\Delta}{1 - w_e\eta}\right\}(1 - w_e\eta). \end{aligned}$$

Since

$$\min\left\{1, \frac{w_e\Delta}{1 - w_e\eta}\right\} = \begin{cases} 1, & w_e(\eta + \Delta) \geq 1 \\ \frac{w_e\Delta}{1 - w_e\eta} & \text{o.w.} \end{cases},$$

we get that

$$\mathbf{Pr}[e \in E_s(\eta + \Delta)] = \min\{1, w_e(\eta + \Delta)\}.$$

□

We now (almost) mirror the definition of $q_\kappa(e)$ for the randomized spanner construction for use in the analysis

DEFINITION 15. *Let $q^*_e(\kappa)$ denote the probability of including edge $e \in E$ in H in Algorithm 1, i.e.*

$$q^*_\kappa(e) = \mathbf{Pr}[e \in E_H(R)].$$

We can now prove

LEMMA 16. *For all $e \in E$*

$$q_\kappa(e) \leq 2q^*_\kappa(e).$$

PROOF. Fix an edge $e \in E$ and let $\eta^* = q_\kappa(e)$. We first note that by Lemma 13, the probability that e is added to H if e is sampled at time η' is only larger than the probability that e is added if it appears at time $\eta > \eta'$. Thus, we have

$$q^*_\kappa(e) \geq \mathbf{Pr}[e \in E_H(\eta^*)] \geq \mathbf{Pr}[e \in E_s(\eta^*)]/2 = q_\kappa(e)/2. \quad (6)$$

□

Hence, we get

LEMMA 17.

$$\sum_{e \in E} w_e q_\kappa(e) \leq 2n^{1+O(1/\kappa)}$$

PROOF. One has $\sum_{e \in E} w_e q^*(e) = \mathbf{E}[|E(H)|]$, where H is the random spanner constructed by Algorithm 1. By construction H does not have cycles of length less than κ, and hence $|E(H)| \leq n^{1+O(1/\kappa)}$ (see, e.g. [8]). Now the result follows by Lemma 16. □

Now the proof of Lemma 2 follows by putting together Lemma 16 and Lemma 12. We now proceed to give an algorithm for efficiently obtaining estimates $\hat{q}_\kappa(u,v)$ of $q_\kappa(u,v)$ guaranteed by Lemma 3.

3.1 Estimating $q_\kappa(e)$

We now show how obtain surrogate values $\hat{q}_\kappa(e)$ that we will use in place of $q_e(\kappa)$ such that $\hat{q}_\kappa(e) \geq R_e/\kappa^2$ and $\sum_{e \in E} \hat{q}_\kappa(e) \leq n^{1+O(1/\kappa)}$ in $\tilde{O}(m)$ time. It will be convenient to introduce another parameter $\delta \in (0,1)$ and write $\hat{q}_{\kappa,\delta}(e)$. Intuitively, δ is the probability that the endpoints of e are more than κ apart a random sample of edges of G where each edge is present independently with probability $\hat{q}_{\kappa,\delta}(e)$, so that $\hat{q}_{\kappa,1/2}(e)$ is an estimate for $q_\kappa(e)$.

The estimation procedure is quite simple. We consider samples of edges of G at a geometric sequence of rates, an calculate the distance between the endpoints of each edge $e \in E$ in the random subgraph given by the sample. Independent repetition of such experiments allows us to estimate the sampling threshold for which the distance between the endpoints of e becomes large. Since the distance calculation is not exact, the approximation to effective resistance that the procedure gives suffers an extra κ factor (we will set $\kappa = \log n$ later). The bound on the sum of the estimated connectivities will follow similarly to the proofs above. We now give a formal description of the estimation procedure.

For each $t = 1, \ldots, T$, where $T = \log(n^4 w_{max}/w_{min})$, and $j = 1, \ldots, J$ for $J = 80 \log n/\delta^2$ define sets E^j_t as follows. For each $e \in E$ add e to E^j_t for t between 1 and $1 + \log(w_e/w_{min})$. Then for each e, repeatedly add e to sets E^j_t with a higher value of t while a coin with heads probability of $1 - \epsilon$ comes up heads.

We will use the Thorup-Zwick distance oracles:

THEOREM 18. *[14] Let $G = (V, E)$ be an undirected weighted graph with n vertices and m edges. For any integer $k \geq 1$ the graph G can be preprocessed in $O(kmn^{1/k})$ expected time constructing a data structure of size $O(kn^{1+1/k})$ such that any subsequent distance query can be answered approximately in $O(k)$ time. The approximate distance returned is is of stretch at most $2k - 1$.*

For $t \in [1:T], j \in [1:J]$ and $e \in E$ the estimation algorithm sets $\eta^j_e(t) = 0$ if the distance between the endpoints of e is reported by an appropriate distance oracle on E^j_t to be at most κ^2 and 0 otherwise. The formal description is given in Algorithm 2.

Algorithm 2: $ESTIMATE(G, \kappa, \delta)$

for $j = 1$ to J **do**
  Set $E^j_t \leftarrow \emptyset$ for $t \in [1:T]$.
  For each $e \in E$ add e to E^j_t for t between 1 and $1 + \log(w_e/w_{min})$.
  for $t = 1$ to $T - 1$ **do**
    Add each $e \in E^j_t$ to E^j_{t+1} independently with probability $1 - \epsilon$.
  end for
end for
for $t = 1$ to T **do**
  Construct a Thorup-Zwick distance oracle for E^j_t, $j \in [1:J]$, denoted by O^j_t.
  for $e = (u,v) \in E$ **do**
    if $O^j_t(u,v) > \kappa^2$ **then**
      $\eta^j_e(t) \leftarrow 1$
    else
      $\eta^j_e(t) \leftarrow 0$
    end if
  end for
end for
for $e \in E$ **do**
  $\hat{q}_{\kappa,\delta}(e) \leftarrow 2^{-t}$, where t is the smallest such that $|\{j : \eta^j_e(t) = 1\}| \geq (1 - \delta)J$
end for
return $\hat{q}_{\kappa,\delta}$

The following lemma gives bounds on $\hat{q}_{\kappa,\delta}$ that yield Lemma 3 after setting $\delta = 1/2$. We will need the ability to chose general δ in the next section (where we will also need property 3 from the lemma below).

LEMMA 19. *With high probability for all $e = (u,v) \in E$ one has*

1. $\hat{q}_{\kappa,\delta}(e) \geq \delta R_e/(4\kappa^2)$
2. $\sum_{e \in E} w_e \hat{q}_{\kappa,\delta}(e) \leq 2n^{1+O(1/\kappa)}$
3. *for all $\eta < q_{\kappa,\delta}(e)$ one has* $\mathbf{Pr}[d_{G_\eta}(u,v) > \kappa] \geq 1 - \delta/2$.

PROOF. First note that by the choice of $T = \log\left(\frac{n^4 w_{max}}{w_{min}}\right)$ we have for each $e \in E$ that $\mathbf{Pr}[d_{G_{2^{-T}}}(u,v) \geq \kappa] = 1 - n^{-c}$ for a constant $c > 0$, so whp $\hat{q}_{\kappa,\delta}(e)$ is defined by Algorithm 2.

To prove the first statement, we argue similarly to Lemma 12. We use the characterization

$$C_{uv} = \min_{x \in \mathbb{R}^V : x_s = 1, x_t = 0} \sum_{e=(u,v) \in E} w_e (x_u - x_v)^2 \quad (7)$$

and denote the conductance of $e = (u,v)$ in G_p by

$$C_{uv}(p) = \min_{x \in \mathbb{R}^V : x_s = 1, x_t = 0} \sum_{e=(u,v) \in E_p} (x_u - x_v)^2. \quad (8)$$

As before, we have

$$\mathbf{Pr}[C_{uv}(p) > 2p/(\delta R_e)] \leq \mathbf{Pr}[C_{uv}(p) > (2/\delta)\mathbf{E}[C_{uv}]] \leq \delta/2. \quad (9)$$

Let $t^* \in [1,T]$ be such that $\hat{q}_{\kappa,\delta}(e) = 2^{-t^*}$. Let

$$\alpha := \mathbf{Pr}[d_{2\hat{q}_{\kappa,\delta}(e)}(u,v) \geq 1/\kappa^2]$$

It follows by an application of Chernoff bounds that

$$\mathbf{Pr}[|\{j : \eta_e^j(t^*-1)=1\}| \notin [\alpha-\delta/2, \alpha+\delta/2]J] < e^{-\delta^2 J/16} = n^{-5}.$$

Hence, we have with high probability that $\alpha \leq 1-\delta/2$, i.e.

$$\mathbf{Pr}\left[d_{2\hat{q}_{\kappa,\delta}(e)}(u,v) \leq \kappa^2\right] \geq \delta/2. \quad (10)$$

Using Rayleigh's monotonicity principle and the fact that the conductance of a path of length at most κ^2 is at least $1/\kappa^2$, we get, combining (10) with (9)

$$(2/\delta)(2\hat{q}_{\kappa,\delta}(e))/R_e \geq 1/\kappa^2, \quad (11)$$

i.e. $q_{\kappa,\delta}(e) \geq \delta R_e/(4\kappa^2)$.

To prove the second inequality, we argue similarly to Lemma 16. Consider a randomized spanner construction process in Algorithm 1 in which an edge e is added to the spanner if there is no path of length smaller than κ in the spanner constructed so far. Since $O_t^j(u,v) \geq \kappa^2$, we have, using the assumption that O_j^i is a κ-approximate distance oracle, that the distance between u and v in the appropriate sampled graph is larger than κ. Thus, similarly to Lemma 16, we have that the probability of including an edge e in the randomized spanner construction is at least $\hat{q}_\kappa(e)/2$. Since the spanner construction process terminates with a graph without cycles of length smaller than κ, we have $\sum_{e\in E} w_e\hat{q}_\kappa(e) \leq 2n^{1+O(1/\kappa)}$. The third inequality follows from the fact the O_t^j is a κ-approximate distance oracle and Chernoff bounds as above. □

Algorithm 2 immediately yields a simple algorithm for spectral sparsification:

Algorithm 3: Spectral sparsification via robust connectivities

1: Set $\hat{q} \leftarrow ESTIMATE(G, \log n, 1/2) \in E$.
2: Sample each $e \in E$ independently with probability $r_e := \min\{1, (cw_e\hat{q}(e)\log^3 n)/\epsilon^2\}$ for a constant c, giving e weight $1/r_e$ if sampled.

We now obtain

THEOREM 20. *Algorithm 1 produces a spectral sparsifier of G with $O(n\log^3 n/\epsilon^2)$ edges whp.*

PROOF. It follows from Lemma 19 that $\hat{q}_{\kappa,1/2}(e)(2\log^2 n) \geq R_e$, where R_e is the effective resistance of e. Hence, the sampled graph is a spectral sparsifier whp by Corollary 7. By Lemma 19

$$\sum_{e\in E} w_e\hat{q}_{\kappa,1/2}(\log n) = O(n^{1+O(1/\log n)}) = O(n).$$

Hence, the size of the sample is bounded by $O(n\log^3 n/\epsilon^2)$. □

In the next section we show that in fact a spectral sparsifier can obtained by taking a union of black-box spanners of random subgraphs of G, thus proving Theorem 4.

4. SPARSIFICATION BY SPANNERS

In this section we give an algorithm for obtaining a spectral sparsifier of an undirected weighted graph G using black box invocations of a spanner construction algorithm, proving Theorem 4. The main challenge here is to overcome dependencies in the sampling process that arise from using a black-box spanner construction. In order to do that, we introduce a procedure that, given a vector of target sampling probabilities $q(e) := \hat{q}_{\kappa,\epsilon}(e), e \in E$, samples edges of G using the black-box spanner construction such that the sampling process (a) stochastically dominates the process of sampling edges independently with probabilities $(1-\epsilon)\min\{1, w_eq(e)\}$ and (b) is stochastically dominated by the process that samples edges independently with probability $\min\{1, w_eq(e)\}$. The procedure consists of a logarithmic number invocations of a basic sampling scheme that we now define.

We first define a sequence of sets $E_1, E_2, \ldots, E_H$ with $H = \log_{1/(1-\epsilon)}\left(\frac{n^4 w_{max}}{w_{min}}\right)$ such that $E_i \cap E_j = \emptyset$ for $i \neq j$ and $\bigcup_{i=1}^H E_i = E$. For each $e \in E$ with weight $w_e \in w_{min}\cdot((1-\epsilon)^{-j}, (1-\epsilon)^{-(j+1)}]$ assign e to set E^j, and for each edge e independntly, keep moving e to higher levels E^i, $i \geq j$, one level at a time while a coin with heads probability $(1-\epsilon)$ comes up heads. Thus, we have $\mathbf{Pr}[e \in E_i \text{ for some } i \geq j] = \min\{1, \frac{w_e}{w_{min}}(1-\epsilon)^j\}$. For $e \in E$ let the *level* of e, denoted by $l(e)$, be the unique index j such that $e \in E_j$, and let $l^*(e)$ denote the smallest j such that $w_{min}(1-\epsilon)^j \leq \min\{1, w_eq(e)\}$.

Let $q(e) = \hat{q}_{\kappa,\epsilon}(e), e \in E$ be the vector of sampling parameters. We first define

Algorithm 4: SAMPLE-SPANNER(G, q)

1: **for** $j = 1, \ldots, H$ **do**
2: Construct a spanner S_j on (V, E_j) of stretch at most κ.
3: For each $e \in S_j$, assign weight 0 to e if $l(e) < l^*(e)$, otherwise assign weight $1/q(e)$.
4: **end for**
5: Return the weighted collection $S_1 \cup \ldots \cup S_H$.

The sampling process takes form

Algorithm 5: SPANNER-SPARSIFY(G, q, ϵ, Z)

1: $q \leftarrow$ ESTIMATE($G, \log n, \epsilon$).
2: $Z \leftarrow \Theta(\log^3 n/((1-\epsilon)\epsilon^3))$
3: **for** $t = 1, \ldots, Z$ **do**
4: $X_t \leftarrow$ SAMPLE-SPANNER(G, q)
5: **end for**
6: **return** $\frac{1}{Z}(X_1 + \ldots + X_H)$

We now prove the main property of our sampling process.

LEMMA 21. *Let $q = \hat{q}_{\kappa,\epsilon}$. Then for $Z = \Omega(\log n)$ whp, i.e. except for a negligible part of the sample space, the sampling process in Algorithm 5 is stochastically dominated by the process of independently sampling an edge e with probability $\min\{1, w_eq(e)\}$ Z times (process B), and dominates the process of independently sampling each edge with probability $(1-\epsilon)\cdot\min\{1, w_eq(e)\}$ Z times(process A).*

PROOF. Denote the spanner constructed by t-th invocation of SAMPLE-SPANNER at level $j \in [1:H]$ by $S_{t,j}$. We denote the system of sets $E_1, \ldots, E_H$ in the t-th invocation of SAMPLE-SPANNER by E_j^t. For an edge $e \in E$ we write $l_t(e)$ to denote the level of e in E_t^j. We refer to an edge $e = (u,v) \in E$ as *free* at level j in invocation t if $d_{E_j^t}(u,v) > \kappa$. By Lemma 19, (3) we have that if $l_t(e) \geq l^*(e)$, then

$$\mathbf{Pr}[e \text{ is free in } S_{t,l_t(e)}] \geq \mathbf{Pr}[d_{G_{\hat{q}_{\kappa,\epsilon}(e)}}(u,v) > \kappa] \geq 1-\epsilon,$$

where the probability is over the coin tosses determining the distribution of $e \in E$ among E_j^t.

Since the invocations of SAMPLE-SPANNER use independent randomness, we have by an application of Chernoff bounds, using the assumption that $Z = \Omega(\log n)$, that with probability at least, say, $1-n^{-5}$ for each $e \in E$ such that $l_t(e) \geq l^*(e)$, one has that e is free in at least a $1-\epsilon$ fraction of the spanners $S_{t,l_t(e)}, t = 1, \ldots, Z$.

Consider any set $W \subseteq E$ of edges. Each edge may be sampled by our process between 0 and Z times. We will show that for any $\{z_e \in [0:Z] | e \in W\}$

$$\prod_{e\in W} ((1-\epsilon)q_\kappa)^{z_e} \leq \mathbf{Pr}[e \text{ is chosen } z_e \text{ times } \forall e \in W] \leq \prod_{e\in W} q_\kappa^{z_e}, \tag{12}$$

where we say that edge e is *chosen* if it is included in a spanner *and given positive weight*.

Indeed, consider the following sampling process. Take Z copies of the edge set E and for each edge first toss a coin to determine which of the Z spanner constructions it belongs to, and then toss coins to determine the level E_j that the edge belongs to. Note that since, by the argument above, whp each edge is free in at least a $(1-\epsilon)$ fraction of the spanners, with probability at least $(1-\epsilon)$ the edge belongs to a spanner construction in which it is free. In that case the edge necessarily belongs to the spanner S_j (since its endpoints are at distance greater than κ in $E_j \setminus \{e\}$), and hence is chosen with probability at least $(1-\epsilon)q_\kappa$. Since the coin tosses for different edges are independent, we get that

$$\mathbf{Pr}[e \text{ is chosen } z_e \text{ times } \forall e \in W] \geq \prod_{e\in W} ((1-\epsilon)q_\kappa)^{z_e}.$$

On the other hand, the weight of an edge is 0 when the edge is at level larger than $l^*(e)$, and hence, since these coin tosses are independent for different edges, for any $W \subseteq E$,

$$\mathbf{Pr}[e \text{ is chosen } z_e \text{ times } \forall e \in W] \leq \prod_{e\in W} q_\kappa^{z_e}.$$

□

We have shown that our sampling process is sandwiched between two independent sampling processes which sample edges with probabilities $(1-\epsilon)Z \cdot \min\{1, w_e q_\kappa\}$ (process A) and $Z \cdot \min\{1, w_e q_\kappa\}$ respectively (process B).

We now obtain

Proof of Theorem 4: Set $Z = O(\log^3 n/((1-\epsilon)\epsilon^3))$. By Lemma 19 we have that

$$\min\{1, Z \cdot w_e \hat{q}_{\kappa,\epsilon}(e)\} \geq \Theta(\log n/\epsilon^2) \cdot \min\{1, w_e R_e\}$$

for all $e \in E$. Let G'_A denote the subgraph sampled by process A and let G'_B denote the subgraph sampled by process B, where process A gives weight $\frac{1}{(1-\epsilon)q(e)}$ to a chosen edge and B gives weight $\frac{1}{q(e)}$. Let G' denote the output of Algorithm 5. Then we have by Corollary 7

$$G'_A \in (1 \pm \epsilon)G, G'_B \in (1 \pm \epsilon)G$$

and by Lemma 21 we have

$$(1-\epsilon) \cdot G'_A \prec G' \prec G'_B,$$

and hence $G' \in (1 \pm O(\epsilon))G$. □

5. REFERENCES

[1] I. Althöfer, G. Das, D. P. Dobkin, D. Joseph, and J. Soares. On sparse spanners of weighted graphs. *Discrete & Computational Geometry*, 9:81–100, 1993.

[2] S. Baswana. Streaming algorithm for graph spanners - single pass and constant processing time per edge. *Inf. Process. Lett.*, 106(3):110–114, 2008.

[3] S. Baswana, T. Kavitha, K. Mehlhorn, and S. Pettie. Additive spanners and (alpha, beta)-spanners. *ACM Transactions on Algorithms*, 7(1), 2010.

[4] S. Baswana and S. Sen. A simple linear time algorithm for computing a (2k-1)-spanner of o(n$^{1+1/k}$) size in weighted graphs. *ICALP*, pages 384–296, 2003.

[5] S. Baswana and S. Sen. Approximate distance oracles for unweighted graphs in expected $o(n^2)$ time. *ACM Transactions on Algorithms*, 2(4):557–577, 2006.

[6] J. D. Batson, D. A. Spielman, and N. Srivastava. Twice-ramanujan sparsifiers. *STOC*, pages 255–262, 2009.

[7] A. A. Benczúr and D. R. Karger. Approximating *s-t* minimum cuts in $\tilde{O}(n^2)$ time. *STOC*, pages 47–55, 1996.

[8] B. Bollobas. *Modern graph theory*. Springer, 1998.

[9] W. S. Fung, R. Hariharan, N. J. A. Harvey, and D. Panigrahi. A general framework for graph sparsification. *STOC*, pages 71–80, 2011.

[10] I. Koutis, G. L. Miller, and R. Peng. Approaching optimality for solving sdd linear systems. *FOCS*, pages 235–244, 2010.

[11] M. Mendel and A. Naor. Ramsey partitions and proximity data structures. *FOCS*, pages 109–118, 2006.

[12] M. Najork, R. Panigrahy, and Y. Xie. How user behavior is related to social affinity. *WSDM*, 2012.

[13] D. Spielman and N. Srivastava. Graph sparsification by effective resistances. *STOC*, pages 563–568, 2008.

[14] M. Thorup and U. Zwick. Approximate distance oracles. *STOC*, 2001.

[15] M. Thorup and U. Zwick. Spanners and emulators with sublinear distance errors. *SODA*, pages 802–809, 2006.