COMPSCI 638: Graph Algorithms September 27, 2019

Lecture 10

Lecturer: Debmalya Panigrahi *Scribe: Kevin Sun*

1 Overview

In the last lecture, we saw that uniform sampling allows us to build a sparsifier of an undirected graph, but when the minimum cut value is small, the sparsifier might be dense. In this lecture, we further study non-uniform sampling for graph sparsification.

2 Sampling via Edge Strengths

Recall the graph sparsification setting: we are given an undirected graph $G = (V, E)$ and some $\epsilon \in (0, 1)$. Our goal is to construct a graph H on V that, with high probability, preserves the value of every cut in G up to a $(1 \pm \epsilon)$ factor. In Lecture 9, we showed that sampling each edge e with probability $p = c \cdot \frac{\log n}{\epsilon^2 \lambda}$ (and setting the weight of sampled edges to $1/p$) yields a valid sparsifier, but when λ is small, it may contain many edges.

To mitigate this, we first recall a couple of definitions. Let λ_e denote the *connectivity* of an edge $e = \{u, v\}$; it is defined as the minimum u-v cut value. Also, let s_e denote the *strength* of an edge e; it is defined as the largest minimum cut value of any induced subgraph containing e. In other words, if $s_e = s$, then there exists a subset of vertices whose induced subgraph has minimum cut value s. In Lecture 9, we showed that $s_e \leq \lambda_e$.

The natural generalization of sampling at rate $p = c \cdot \frac{\log n}{\epsilon^2 \lambda}$ would be to replace the λ with λ_e for each edge e. However, as we will see, this approach is challenging to analyze. Instead, we will present the following result of Benczúr and Karger [BK15].

Algorithm 1 Sampling via Edge Strengths (Benczúr and Karge [BK15])

Input: An undirected graph $G = (V, E)$ on n vertices with edge strengths s_e.

We will construct a graph H with vertex set V and edge set E_H.
for $e \in E$ **do**
    Add e to E_H with probability $p = c \cdot \frac{\log n}{\epsilon^2 s_e}$ for some constant c.
    if e was added to E_H **then**
        Set the weight of e in H to be $1/p_e$.
return $H = (V, E_H)$ as a sparsifier for G

Theorem 1 (Benczúr and Karger [BK15])**.** *The output of Algorithm 1, with high probability, preserves the value of every cut up to a $(1 \pm \epsilon)$ factor. The expected number of edges it contains is $O(n \log n / \epsilon^2)$.*

We first bound the expected number of edges in the sparsifier.

Lemma 2. *In any undirected graph $G = (V, E)$ on n vertices,*

$$\sum_{e \in E} \frac{1}{\lambda_e} \leq \sum_{e \in E} \frac{1}{s_e} \leq n - 1$$

Proof. Note that the first inequality follow from the inequality $s_e \leq \lambda_e$, which we proved in Lecture 9. We prove the second inequality by induction on the number of vertices in the graph.

Let C denote the set of edges in a minimum cut, and let $\lambda = |C|$. Notice that any edge $e \in C$ satisfies $\lambda_e = \lambda$. Furthermore, for any $e \in C$, the graph G itself is an induced subgraph containing e, and G has minimum cut value λ. Thus, $s_e \geq \lambda = \lambda_e$, which means $s_e = \lambda_e = \lambda$, so

$$\sum_{e \in C} \frac{1}{s_e} = \lambda \cdot \frac{1}{\lambda} = 1. \tag{1}$$

Now suppose we remove the edges of C, creating graphs G_1 and G_2 on n_1 and n_2 vertices, respectively (so $n_1 + n_2 = n$). Every remaining edge e has some new strength s'_e, and $s'_e \leq s_e$ because any induced subgraph in G_1 (or G_2) is also an induced subgraph in G with the same minimum cut value. Applying the induction hypothesis on G_1, we have

$$\sum_{e \in E(G_1)} \frac{1}{s_e} \leq \sum_{e \in E(G_1)} \frac{1}{s'_e} \leq n_1 - 1,$$

and the same statement holds for G_2. Putting this together with (1), we get

$$\sum_{e \in E} \frac{1}{s_e} = \sum_{e \in E(G_1)} \frac{1}{s_e} + \sum_{e \in E(G_2)} \frac{1}{s_e} + \sum_{e \in C} \frac{1}{s_e} \leq (n_1 - 1) + (n_2 - 1) + 1 = n - 1. \qquad \square$$

Notice that Lemma 2 implies the second part of Theorem 1. Furthermore, this bound is tight up to a logarithmic factor since any sparsifier of a connected graph on n vertices must contain at least $n - 1$ edges to remain connected. So for the remainder of this section, we will prove that with high probability, every cut is preserved up to a $(1 \pm \epsilon)$ factor.

A simplifying assumption: For ease of presentation, we assume that every edge has strength $s_e = 2^i \cdot \lambda$ for some integer $i \geq 0$, where λ is the size of the global minimum cut. Note that this assumption is fairly simple to justify: if some strength value s_e is between $2^k\lambda$ and $2^{k+1}\lambda$ for some k, we can mentally boost its value to $2^{k+1}\lambda$ and adjust for this by similarly boosting the constant c in our sampling rate by a factor of two.

We begin by observing some structural properties that edge strengths induce on a graph. Let k be a positive integer and let us define a relation on V as follows:

$$R_k = \{(u, v) : s_{u,v} \geq k\}.$$

In other words, two vertices u and v are related under R_k if there exists an induced subgraph that contains u and v with minimum cut at least k. (Note that the "strength" of $\{u, v\}$ is well-defined even if $\{u, v\}$ is not an edge in the graph.)

Lemma 3. *For every k, the relation R_k on V is an equivalence relation.*

Proof. The relation is trivially reflexive, and it is symmetric because the graph is undirected.

Now suppose uR_kv and vR_kw for some vertices $u, v, w \in V$. Then there is a subgraph G_1 that contains u, v with minimum cut at least k, and a subgraph G_2 that contains u, v with minimum cut at least k. Consider the graph G_3 induced by the union of the vertices in G_1 and G_2. Now consider any cut with edge set C in G_3: since v is in both G_1 and G_2, C must be a non-trivial cut at least one of G_1, G_2. Thus, C must contain at least k edges, so the minimum cut size of G_3 is at least k. □

Now recall that if R is an equivalence relation on V, then R induces a partition on V; we let $P(R)$ denote the subsets, also known as blocks, created by this partition. Thus, Lemma 3 tells us that if u, v are in a block of R_k, then the strength of $\{u, v\}$ is at least k. Furthermore, the next lemma shows that for any k, the partition $P(R_{k+1})$ is a refinement of $P(R_k)$. In other words, every block of $P(R_{k+1})$ is a subset of some block of $P(R_k)$ (see Fig. 1).

Lemma 4. *For any integer k, the partition $P(R_{k+1})$ is a refinement of the partition $P(R_k)$.*

Proof. Let B be some block of $P(R_{k+1})$ containing a vertex u. It suffices to show that every vertex $v \in B$ is in the same block of $P(R_k)$ as u, that is, u and v are also related under R_k. This is quite straightforward: since u and v are related under R_{k+1}, there exists an induced subgraph containing $\{u, v\}$ with minimum cut value at least $k + 1 > k$. Thus, u and v are related under R_k. □

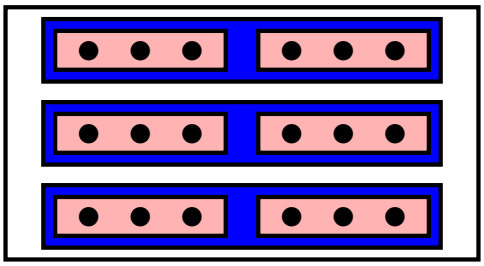

Figure 1: A visual representation of the partitions induced on V. The largest box contains all of the vertices, the blue boxes represent the blocks of $P(R_\lambda)$, and the red boxes represent the blocks of $P(R_{\lambda'})$ for some $\lambda' > \lambda$. Note it is possible for vertices to exist outside of the blue boxes.

The strength of any two vertices is at least λ, so the partition $P(R_\lambda)$ is simply V. Now consider the partition $P(R_{2\lambda})$. If $s_{u,v} < 2\lambda$, then $s_{u,v} = \lambda$ (by our assumption), and u and v must be in different blocks of $P(R_{2\lambda})$. In general, by the definition of R_k, if the strength of $\{u, v\}$ is less than $2^i\lambda$ then the edge $\{u, v\}$ (if it exists) must have endpoints in different blocks of $P(R_{2^i\lambda})$.

This observation allows us to view edge sampling as a process that occurs in phases, rather than all at once. Observe that the following procedure is equivalent to sampling each edge e with probability $p_e = c \cdot \frac{\log n}{\epsilon^2 s_e}$:

1. Sample every edge with probability $p = c \cdot \frac{\log n}{\epsilon^2 \lambda}$.
2. Sample every sampled edge within each block of $P(R_{2\lambda})$ with probability $1/2$.
3. Sample every sampled edge within each block of $P(R_{4\lambda})$ with probability $1/2$.
4. Continue in this manner through the largest strength value.

We now turn to the main portion of the proof. Intuitively, our phase-based sampling procedure samples edges with the same strength at the "proper" sampling rate according to the uniform sampling theorem from Lecture 9. Each phase, with high probability, contributes some small amount of error to our final estimate. We can bound the total error by $O(\epsilon)$, and since there are at most $O(\log n)$ phases, we can apply a union bound on the overall failure probability.

More formally, let C be a cut with μ edges, and let $w_H(C)$ denote the weight of C in the sparsifier H. We will show that, for some constant d,

$$\Pr\left(w_H(C) \notin [1-\epsilon, 1+\epsilon]\mu\right) \leq \frac{1}{n^d}$$

Note that this is equivalent to showing $|w_H(C) - \mu| = O(\epsilon) \cdot \mu$ with high probability, because we can suitably adjust the constant c in our sampling rate.

To bound $|w_H(C) - \mu|$, we will bound the error incurred by each phase of our sampling procedure. In particular, we will find δ_i for $i \geq 0$ such that the error incurred by the i-th phase is at most $\delta_i \mu$, and $\sum_{i=0}^{k} \delta_i = O(\epsilon)$. (Our first phase samples all edges with probability p and incurs error δ_0.) Let C_i denote the set of edges in C with strength at least $2^i\lambda$, and let $\mu_i = |C_i|$. Observe that in the initial phase, we are performing uniform sampling from Lecture 9, so the incurred error is $\delta_0 = \epsilon$ with high probability.

In the next phase, we sample every edge with strength at least 2λ with probability $p_1 = 1/2$. If any such edge exists in C, then in fact $\mu_1 \geq 2\lambda$. Substituting this into a Chernoff bound, we see that the probability of incurring δ_1 error in this phase is at most

$$\exp\left(-\frac{\delta_1^2 p_1 \mu_1}{3}\right) \leq \exp\left(-\frac{\delta_1^2 \cdot 2\lambda}{6}\right) = \frac{1}{n^{d_1}}$$

for some constant d_1 by setting $\delta_1 = \sqrt{\log n / 2\lambda}$. We can apply the same reasoning in the subsequent phase, on edges with strength at least 4λ, and set $\delta_2 = \sqrt{\log n / 4\lambda}$. In general, we can set $\delta_i = \sqrt{\log n / 2^i\lambda}$ for all $i \geq 1$. Thus, the total factor of our error is

$$\delta_0 + \sum_{i=1}^{k} \delta_i = \epsilon + \sum_{i=1}^{k} \sqrt{\frac{\log n}{2^i\lambda}} = \epsilon + O\left(\sqrt{\frac{\log n}{\lambda}}\right),$$

where last equality follows by noticing that δ_i forms a geometric series for $i \geq 1$.

Recall that we want to the δ_i to satisfy $\delta_0 + \cdots + \delta_k = O(\epsilon)$, so we are not done if $\sqrt{\log n/\lambda} > \epsilon$. But if this is the case, then $p = \log n/\epsilon^2\lambda > 1$, which means every edge is sampled in the first phase. In fact, we can extend this reasoning to claim that every edge is sampled in every phase until we reach the first i such that $\log n/2^i\lambda\epsilon^2 < 1$. At this point, every edges with strength less than $2^i\lambda$ has been successfully sampled, so we can safely set

$$\delta_0 = \delta_1 = \cdots = \delta_{i-1} = 0.$$

Finally, in this phase we have $\log n/2^i\lambda\epsilon^2 < 1$, which implies $\sqrt{\log n/2^i\lambda} < \epsilon$, i.e., the ϵ term dominates the error. So we can conclude $\sum_{i=0}^{k} \delta_i = O(\epsilon)$, as desired.

Computing edge strengths: Notice that Algorithm 1 does not specify how we can compute the value of s_e for every edge e. But at the same time, Lemma 2 tells us that we do not actually need the exact value of s_e; it suffices to compute some value $s'_e \leq s_e$ (so the sampling rate can only increase) while maintaining $\sum_e 1/s'_e \leq n-1$. Indeed, Benczúr and Karger [BK15] show how this can be done efficiently by using a maximum spanning tree.

2.1 Sampling via Edge Connectivities

We conclude by briefly discussing graph sparsification via edge connectivities. Fung et al. [FHHP19] showed that if we sample each edge e with probability $p_e = c \cdot \log^2 n / \epsilon^2 \lambda_e$, then the result is indeed a $(1 \pm \epsilon)$-sparsifier with high probability. Note that the result is slightly denser than sampling by strengths (by a $\log n$ factor), but it uses the more natural notion of edge connectivity.

The proof is quite involved, so instead, we give some intuition for why the additional $\log n$ factor helps us achieve the sparsification guarantee. Recall that in the proof of Theorem 1, we used the following fact repeatedly: if a cut C contains an edge with strength at least $2^i\lambda$, then C contains at least $2^i\lambda$ such edges. This allows us to bound the error in the sparsifier against these $2^i\lambda$ edges and ignore the rest of the cut.

In contrast, recall the graph $K(1, 1, n)$ defined in Lecture 9: $V = \{s, v_1, \ldots, v_n, t\}$ and E contains $\{s, v_i\}$ and $\{v_i, t\}$ for every i, as well as $\{s, t\}$. All edges have strength 2, but the connectivity of $\{s, t\}$ is $n + 1$. However, this does not imply that the cut $\{s\}$, say, has $n + 1$ edges with connectivity at least $n + 1$. Thus, we cannot ignore the rest of the cut because we cannot bound our error against this one edge with such high variance.

Instead, Fung et al. [FHHP19] show that by boosting our sampling rate by a $\log n$ factor, we can bound the error in the sparsifier against the cut value in the *entire* graph (rather than within each strength class). In our example, this amounts to bounding the error against the $n + 1$ edges in the cut $\{s\}$ rather than the one edge with connectivity $n + 1$, and we pay an extra $\log n$ factor to do so.

References

[BK15] András A Benczúr and David R Karger. Randomized approximation schemes for cuts and flows in capacitated graphs. *SIAM Journal on Computing*, 44(2):290–319, 2015.

[FHHP19] Wai-Shing Fung, Ramesh Hariharan, Nicholas JA Harvey, and Debmalya Panigrahi. A general framework for graph sparsification. *SIAM Journal on Computing*, 48(4):1196–1223, 2019.