Tight Bounds on Approximating Minimum Cuts in Insertion-Only Streaming Model

Alexandro Garces

DIMACS REU at Rutgers University

agarces2@mit.edu

July 20, 2023

Basic Definitions

Graphs in the Insertion Streaming Model

We begin with a set of nodes $V = \{1, \ldots, n\}$. One by one, we are given a sequence or "stream" of m edges $e = (u, v)$ where $u, v \in V$. The goal is to minimize the amount of space used while the stream is coming in.

Definition: Cut

A cut in a graph is a partition $(S, V \setminus S)$ of the graph into two non-empty sets.

Definition: Minimum Cut

The minimum cut is a cut $(S', V \setminus S')$ with the fewest number of crossing edges.

Motivation

- Computing the value of a minimum cut in a graph is an incredibly important problem.
- The streaming model is a very practical model for real-world situations.
- In real world applications, it is usually sufficient to get a good approximation.

Approximation Definitions

$(1+\epsilon)$-Approximation to Cut Value

Given cut value λ, we say λ' is a $(1+\epsilon)$-approximation to λ if

$$(1-\epsilon)\lambda \leq \lambda' \leq (1+\epsilon)\lambda$$

$(1+\epsilon)$-Cut Sparsifier

Given graph $G = (V, E)$, we say graph $H = (V, E')$ is a $(1+\epsilon)$-cut sparsifier of G if, with high probability,

$$(1-\epsilon)\lambda_G(S, V \setminus S) \leq \lambda_H(S, V \setminus S) \leq (1+\epsilon)\lambda_G(S, V \setminus S)$$

holds $\forall S \subset V, S \neq \emptyset$.

Prior Results

Theorem 1 (Black Box)

There exists a single-pass algorithm $\mathcal{A}$ that returns a for-all $(1+\epsilon)$ cut-sparsifier for a graph with $O(\text{poly } n)$ edge weights in the insertion-only streaming model using $\tilde{O}(\frac{n}{\epsilon^2})$ space. [AGM12]

Theorem 2

Any single-pass algorithm that returns a $(1+\epsilon)$-approximation to the minimum cut problem in the insertion-only streaming model requires at least $\Omega(n \log(1/\epsilon))$ bits. [AG09]

Our Results

Result 1

There exists a single-pass algorithm that returns a $(1+\epsilon)$-approximation to the minimum cut problem on graphs with $O(\text{poly } n)$ edge weights in the insertion-only streaming model using $\tilde{O}(\frac{n}{\epsilon})$ space.

Result 2

Any single-pass algorithm that computes a $(1+\epsilon)$-approximation to the minimum cut of a graph in an insertion-only stream uses at least $\Omega(\frac{n}{\epsilon})$ bits of space.

Important Sparsifier Properties

Mergeability

Let H_1 and H_2 be $(1+\epsilon)$ cut-sparsifiers of graphs G_1 and G_2 on the same vertices. Then, $H_1 \cup H_2$ is a $(1+\epsilon)$ cut-sparsifier of graph $G_1 \cup G_2$.

Composability

Let H be a $(1+\epsilon)$ cut-sparsifier of H_1 and H_1 be a $(1+\epsilon)$ cut-sparsifier of G. Then, H is a $(1+\epsilon)^2 \approx (1+2\epsilon)$ cut-sparsifier of G.

Existing Algorithm [McGregor14]

1. Partition stream into $t = m\epsilon^2/n$ groups, each with n/ϵ^2 edges
2. Let G_i^0 denote the i-th group of edges, recursively define

$$G_i^j = G_{2i-1}^{j-1} \cup G_{2i}^{j-1}$$

3. Build sparsified graphs for each of the G_i^js:

$$H_i^0 = G_i^0, H_i^j = \mathcal{A}(H_{2i-1}^{j-1} \cup H_{2i}^{j-1})$$

where $\mathcal{A}(G)$ denotes running the black box algorithm on G with an approximation value of δ which we will set.

Existing Algorithm (cont.)

From the mergeability and composability properties of sparsifiers, we see that $H_1^{\log t}$ is a $(1+\delta)^{\log t}$ sparsifier of G. Now, suppose we pick $\delta = \frac{\epsilon}{2 \log t}$. Thus,

$$(1+\delta)^{\log t} \leq \exp(\delta \log t) = \exp(\epsilon/2) \leq 1 + \epsilon$$

where the first and second inequalities come from the fact that $1 + x \leq e^x$ for all x and $e^x \leq 1 + 2x$ for $x \in [0, 1]$, respectively.

Thus, we obtain a $(1+\epsilon)$ cut-sparsifier. The space complexity of the algorithm is $\tilde{O}(\frac{n}{\epsilon^2})$.

New Black Box

The space bound of the algorithm above depends entirely on the space bound of the black box. Thus, if we can use a better black box, we can directly improve the algorithm's space bound!

Theorem 3 (Better Black Box)

There exists an algorithm $\mathcal{K}$ that returns a graph H with $\tilde{O}(n/\epsilon)$ edges in $\tilde{O}(mn)$ time that gives a **"for-each"** $(1+\epsilon)$ cut-sparsifier. [CGPSSW18]

"for-all" vs. "for-each" Sparsifiers

"for-all" sparsifiers guarantees that, with high probability, **ALL** cuts will be preserved within a factor of $(1+\epsilon)$.

"for-each" sparsifiers guarantee that **EACH** cut is preserved within a factor of $(1+\epsilon)$ with high probability.

Mergability and composability hold for both sparsifier types!

Problem :(

We can't apply the new black box directly! We need to somehow bound the number of cuts we're interested in looking at to a polynomial amount.

Karger Saves the Day! [KS96]

Definition: α-minimal cut

An α-minimal cut is a cut of value within a multiplicative factor of α of the minimum.

Karger's Cut Counting Lemma

In any graph, the number of α-minimal cuts is $\leq (2n)^{2\alpha}$.

All of these minimal cuts can be obtained with high probability by running Karger's Recursive Contraction Algorithm, which can be done in $O(m \log(n^2/m))$ space.

Our Algorithm

1. Partition stream into $t = m\epsilon/n$ groups, each with n/ϵ edges
2. Let G_i^0 denote the i-th group of edges, recursively define

$$G_i^j = G_{2i-1}^{j-1} \cup G_{2i}^{j-1}$$

3. Build sparsified graphs for each of the G_i^js:

$$H_i^0 = G_i^0, H_i^j = \mathcal{K}(H_{2i-1}^{j-1} \cup H_{2i}^{j-1})$$

   where we run $\mathcal{K}$ on a value $\epsilon' = \frac{\epsilon}{2 \log t}$.
4. Use $\mathcal{A}$ to construct a "for-all" 1.1 cut-sparsifier
5. Retrieve all 1.1-minimal cuts via Recursive Contraction and store them
6. Query $H_1^{\log t}$ for values of all the stored cuts and return the smallest value

Our Algorithm (Analysis)

By the same analysis as before, $H_1^{\log t}$ is a "for-each" $(1+\epsilon)$ cut-sparsifier of G.

The minimum cut λ has value at most 1.1λ in the "for-all" cut sparsifier. By retrieving all 1.1-minimal cuts, we guarantee that we also retrieve the minimum cut. Since $H_1^{\log t}$ is a "for-each" $(1+\epsilon)$ cut-sparsifier, it preserves with high probability, say $1 - 1/n^4$, the value of each cut we're interested in.

Let $\mathcal{C} = \{C_1, \ldots, C_k\}$ be the set of 1.1-minimal cuts returned by the contraction procedure. Let A be the event that we choose the wrong cut in this set to take as the source of the minimum cut value. Denote the event that cut C_i is not preserved as E_{C_i}. Then,

$$Pr(A) \leq Pr(E_{C_1} \cup \ldots \cup E_{C_k}) \leq \sum_{i=1}^{n^{2.2}} Pr(E_{C_i}) = \sum_{i=1}^{n^{2.2}} n^{-4} \leq 1/n$$

Thus, the probability that we approximate the minimum cut correctly is at least $1 - 1/n$.

Our Algorithm (Analysis cont.)

The space bound analysis follows almost immediately from the analysis of the algorithm in McGregor's survey paper. We only need two sparsifiers for each of the $\log t \leq \log n$ levels, and each sparsifier takes $\tilde{O}(n/\epsilon)$ space. Thus, we have an overall space bound of $\tilde{O}(n/\epsilon)$.

Steps Forward

Are similar results possible in dynamic streams? Hopefully so...

Acknowledgement

I am extremely grateful for the DIMACS REU at Rutgers University for giving me the opportunity to perform research. Thank you to NSF grant CNS-2150186 for funding my research!

Thank you very much to my mentors, Professor Sepehr Assadi and Vihan Shah. I couldn't have done this work without them.

References

Kook Jin Ahn, Sudipto Guha, Andrew McGregor

Graph sketches: sparsification, spanners, and subgraphs

PODS 31, 5–14 (2012).

Kook Jin Ahn, Sudipta Guha

Graph Sparsification in the Semi-streaming Model

ICALP 328–338 (2009).

Andrew McGregor

Graph stream algorithms: a survey

SIGMOD Record 43(1), 9–20 (2014).

References

Timothy Chu, Yu Gao, Richard Peng, Sushant Sachdeva, Saurabh Sawlani, Junxing Wang

Graph Sparsification, Spectral Sketches, and Faster Resistance Computation, via Short Cycle Decompositions

FOCS 59th Annual Symposium (2018).

David Karger, Clifford Stein

A New Approach to the Minimum Cut Problem

Journal of the ACM 43 (1996).

Xiaoming Sun, David P. Woodruff

Tight bounds for graph problems in insertion streams

APPROX/RANDOM (2015).

The End!

Thanks for listening :)