Electronic Colloquium on Computational Complexity, Report No. 108 (2020)

Query Complexity of Global Minimum Cut

Arijit Bishnu* Arijit Ghosh† Gopinath Mishra‡ Manaswi Paraashar§

Abstract

In this work, we resolve the query complexity of global minimum cut problem for a graph by designing a randomized algorithm for approximating the size of minimum cut in a graph, where the graph can be accessed through local queries like Degree, Neighbor, and Adjacency queries.

Given $\epsilon \in (0,1)$, the algorithm with high probability outputs an estimate $\hat{t}$ satisfying the following $(1-\epsilon)t \leq \hat{t} \leq (1+\epsilon)t$, where m is the number of edges in the graph and t is the size of minimum cut in the graph. The expected number of local queries used by our algorithm is $\min\left\{m+n, \frac{m}{t}\right\} \text{poly}\left(\log n, \frac{1}{\epsilon}\right)$ where n is the number of vertices in the graph. Eden and Rosenbaum showed that $\Omega(m/t)$ many local queries are required for approximating the size of minimum cut in graphs. These two results together resolve the query complexity of the problem of estimating the size of minimum cut in graphs using local queries.

Building on the lower bound of Eden and Rosenbaum, we show that, for all $t \in \mathbb{N}$, $\Omega(m)$ local queries are required to decide if the size of the minimum cut in the graph is t or $t-2$. Also, we show that, for any $t \in \mathbb{N}$, $\Omega(m)$ local queries are required to find all the minimum cut edges even if it is promised that the input graph has a minimum cut of size t. Both of our lower bound results are randomized, and hold even if we can make Random Edge query apart from local queries.

1 Introduction

Global minimum cut (denoted MinCut) for a connected, unweighted, undirected and simple graph $G=(V,E)$, $|V|=n$ and $|E|=m$, is a partition of the vertex set V into two sets S and $V \setminus S$ such that the number of edges between S and $V \setminus S$ is minimized. Let Cut(G) denote this edge set corresponding to a minimum cut in G, and t denote $|\text{Cut}(G)|$. The problem is so fundamental that researchers keep coming back to it again and again across different models [McG14, KS93, Kar93, KT19, MN20, AGM12]. Fundamental graph parameter estimation problems, like estimation of the number of edges [Fei06, GR08], triangles [ELRS17], cliques [ERS18], stars [GRS11], etc. have been solved in the local and bounded query models [GGR98, GR08, KKR04]. Estimation of the size of MinCut is also in the league of such fundamental problems to be solved in the model of local queries.

In property testing [Gol17], a graph can be accessed at different granularities — the query oracle can answer properties about graph that are local or global in nature. Local queries involve the relation of a vertex with its immediate neighborhood, whereas, global queries involve the relation between sets of vertices. Recently using a global query, named Cut Query [RSW18], the problem of estimating and finding MinCut was solved, but the problem of estimating or finding MinCut

*Indian Statistical Institute, Kolkata, India. E-mail: arijit@isical.ac.in
†Indian Statistical Institute, Kolkata, India. E-mail: arijitiitkgpster@gmail.com
‡Indian Statistical Institute, Kolkata, India. E-mail: gopianjan117@gmail.com
§Indian Statistical Institute, Kolkata, India. E-mail: manaswi.isi@gmail.com

ISSN 1433-8092

using local queries has not been solved. The fundamental contribution of our work is to resolve the query complexity of MinCut using local queries. We resolve both estimating and finding variants of the problem. To start with, we formally define the query oracle models we would be needing for discussions that follow.

The query oracle models. We start with the most ubiquitous local queries and the random edge query for a graph $G = (V, E)$ where the vertex set V is known but the edge set E is unknown.

- Local Query
  - Degree query: given $u \in V$, the oracle reports the degree of u in V;
  - Neighbor query: given $u \in V$, the oracle reports the i-th neighbor of u, if it exists; otherwise, the oracle reports $\perp$;
  - Adjacency query: given $u, v \in V$, the oracle reports whether $\{u, v\} \in E$.
- Random Edge query: The query outputs an uniformly random edge of G.

Apart from the above local queries, in the last few years, researchers have also used the Random Edge query [ABG$^+$18, AKK19]. Notice that the randomness will be over the probability space of all edges, and hence, a random edge query is not a local query. We use this query in conjunction with local queries only for lower bound purposes. The other query oracle relevant for our discussion will be a *global query* called the Cut Query proposed by Rubinstein et al. [RSW18] that was motivated by submodular function minimization. The query takes as input a subset S of the vertex set V and returns the size of the cut between S and $V \setminus S$ in the graph G.

Prologue. Our motivation for this work is twofold — MinCut is a fundamental graph estimation problem that needs to be solved in the local query oracle model and the lower bound of Eden and Rosenbaum [ER18] who extended the seminal work of Blais et al. [BBM11] to develop a technique for proving query complexity lower bounds for graph properties via reductions from communication complexity. Using those techniques, for graphs that can be accessed by only local queries like Degree, Neighbor, Adjacency and Random Edge, Eden and Rosenbaum [ER18] showed that MinCut has a lower bound of $\Omega(m/t)$, where m and t are the number of edges and the size of the minimum cut, respectively, in the graph. In this work, we show that the query complexity of estimating MinCut using local queries only (and not Random Edge) is $\widetilde{\Theta}\left(m/t\right)$, thus proving a matching upper bound. For designing the query algorithm for MinCut that matches the lower bound, we revisit the fundamental work of Karger [Kar93]. The power of query oracles allows us to use an ingenious coupling of a guessing scheme with Karger's result of upper bounding the number of cuts of a particular size, to come up with the algorithm.

Prior to our work, no local query based algorithm has been developed for MinCut. But it was Rubinstein et al. [RSW18] who studied MinCut for the first time using Cut Query, a global query. They showed that there exists a randomized algorithm for finding a MinCut in G using $\widetilde{\mathcal{O}}(n)$[1] many Cut Query. Graur et al. [GPRW20] showed a matching (deterministic) lower bound for finding MinCut using Cut Query.

Problem statements and results. We focus on two problems in this work.

[1] $\widetilde{\mathcal{O}}(n)$ hides polylogarithmic terms in n.

Minimum Cut Estimation **Input:** A parameter $\epsilon \in (0, 1)$, and access to an unknown graph G via local queries **Output:** An $(1 \pm \epsilon)$-approximation to $|\textsc{Cut}(G)|$.

Minimum Cut Finding **Input:** Access to an unknown graph G via local queries **Output:** Find a set $\textsc{Cut}(\mathrm{G})$.

Our results are the following.

Theorem 1. (Minimum cut estimation using local queries) *There exists an algorithm, with* Degree *and* Neighbor *query access to an unknown graph $G = (V, E)$, that solves the minimum cut estimation problem with high probability. The expected number of queries used by the algorithm is*

$$\min\left\{m + n, \frac{m}{t}\right\} \mathrm{poly}\left(\log n, \frac{1}{\epsilon}\right).$$

Notice that Theorem 1 coupled with the matching lower bound result of Eden and Rosenbaum [ER18] closes the MinCut estimation problem in graphs using local queries.

Building on the lower bound construction of Eden and Rosenbaum [ER18], we show that no nontrivial query algorithm exists for finding a minimum cut or even estimating the exact size of a minimum cut in graphs.

Theorem 2. (Lower bound for minimum cut finding, i.e., $\textsc{Cut}(\mathbf{G})$) *Let $m, n, t \in \mathbb{N}$ with $t \leq n - 1$ and $2nt \leq m \leq \binom{n}{2}$. Any algorithm that has access to* Degree*,* Neighbor*,* Adjacency *and* Random Edge *queries to an unknown graph $G = (V, E)$ must make at least $\Omega(m)$ queries in order to find all the edges in a minimum cut of G with probability $2/3$.*

Theorem 3. (Lower bound for finding the exact size of the minimum cut, i.e., $|\textsc{Cut}(G)|$) *Let $m, n, t \in \mathbb{N}$ with $2 \leq t \leq n - 2$ and $2nt \leq m \leq \binom{n}{2}$. Any algorithm that has access to* Degree*,* Neighbor*,* Adjacency *and* Random Edge *queries to an unknown graph $G = (V, E)$ must make at least $\Omega(m)$ queries in order to decide whether $|\textsc{Cut}(G)| = t$ or $|\textsc{Cut}(G)| = t - 2$ with probability $2/3$.*

Local queries show a clear separation in its power in finding MinCut as opposed to the estimation problem. This is established by using the tight lower bound of minimum cut estimation (viz. $\Omega(m/t)$ lower bound of Eden and Rosenbaum and our Theorem 1) vis-a-vis minimum cut finding as mentioned in our Theorems 2 and 3 on lower bound for finding $\textsc{Cut}(\mathrm{G})$.

Notations. In this paper, we denote the set $\{1, \ldots, n\}$ by $[n]$. For ease of notation we sometimes use $[n]$ to denote the set of vertices of a graph. We say $x \geq 0$ is an $(1 \pm \epsilon)$-approximation to $y \geq 0$ if $|x - y| \leq \epsilon y$. $V(G)$ and $E(G)$ would denote the vertex and edge sets when we want to make the graph G explicit, else we use V and E. For a graph G, $\textsc{Cut}(G)$ denotes the set of edges in a minimum cut of G. Let A_1, A_2 be a partition of V, i.e., $V = A_1 \cup A_2$ with $A_1 \cap A_2 = \emptyset$. Then, $\mathcal{C}_G(A_1, A_2) = \{\{u, v\} \in E \,:\, u \in A_1 \text{ and } v \in A_2\}$. The statement *with high probability* means that the probability of success is at least $1 - \frac{1}{n^c}$, where c is a positive constant. $\widetilde{\Theta}(\cdot)$ and $\widetilde{\mathcal{O}}(\cdot)$ hides a $\mathrm{poly}\left(\log n, \frac{1}{\epsilon}\right)$ term in the upper bound.

Organization of the paper Section 2 discusses the query algorithm for estimating the MinCut while Section 3 proves lower bounds on finding the MinCut. Section 4 concludes with a few observations.

2 Estimation algorithm

In this Section, we will prove Theorem 1. In Section 2.1, we talk about the intuitions and give the overview of our algorithm. We formalize the intuitions in Section 2.2.

2.1 Overview of our algorithm

We start by assuming that a lower bound $\hat{t}$ on $t = |\textsc{Cut}(G)|$ is known. Later, we discuss how to remove this assumption.

We generate a random subgraph H of G by sampling each edge of the graph G independently with probability $p = \Theta\left(\log n/\epsilon^2\hat{t}\right)$. Using Chernoff bound, we can show that any particular cut of size k, $k \geq t$, in G is *well approximated* in H with probability at least $n^{-\Omega(k/\hat{t})}$. With this idea, consider the following Algorithm, stated informally, for minimum cut estimation.

Algorithm-Sketch (works with $\hat{t} \leq t$)

Step-1: Generate a random subgraph H of G by sampling each edge in G independently with probability $p = \Theta\left(\log n/\epsilon^2\hat{t}\right)$. Note that H can be generated by using $\widetilde{O}\left(m/\hat{t}\right)$ many Degree and Neighbor queries in expectation. We will discuss it in Algorithm 1 in Section 2.2.

Step-2 Determine $|\textsc{Cut}(H)|$ and report $\widetilde{t} = \frac{|\textsc{Cut}(H)|}{p}$ as an $(1 \pm \epsilon)$-approximation of $|\textsc{Cut}(G)|$.

The number of queries made by the above algorithm is $\widetilde{O}\left(m/\hat{t}\right)$ in expectation. But it produces correct output only when the vertex partition corresponding to $\textsc{Cut}(G)$ and $\textsc{Cut}(H)$ are the same. This is not the case always. If we can show that all cuts in G is approximately preserved in H, then Algorithm-Sketch produces correct output with high probability. The main bottleneck to prove it is that the total number of cuts in G can be exponential. A result of Karger (stated in the following lemma) will help us to make Algorithm-Sketch work.

Lemma 4 (Karger [Kar93])**.** *For a given graph G the number of cuts in G of size at most $j \cdot |\textsc{Cut}(G)|$ is at most n^{2j}.*

Using the above lemma along with Chernoff bound, we can show the following.

Lemma 5. *Let G be a graph, $\hat{t} \leq t = |\textsc{Cut}(G)|$ and $\epsilon \in (0,1)$. If $H(V(G), E_p)$ be a subgraph of G where each edge in $E(G)$ is included in E_p with probability $p = \min\left\{\frac{200\log n}{\epsilon^2\hat{t}}, 1\right\}$ independently, then every cut of size k in G has size $pk(1 \pm \epsilon)$ in H with probability at least $1 - \frac{1}{n^{10}}$.*

The above lemma implies the correctness of Algorithm-Sketch, which is for minimum cut estimation when we know a lower bound $\hat{t}$ of $|\textsc{Cut}(G)|$. But in general we do not know any such $\hat{t}$. To get around the problem, we start guessing $\hat{t}$ starting from $\frac{n}{2}$ each time reducing $\hat{t}$ by a factor of 2. The guessing scheme gives the desired solution due to Lemma 5 coupled with the following intuition when $\hat{t} = \Omega(t \log n/\epsilon^2)$ — if we generate a random subgraph H of G by sampling each edge with probability $p = \Theta\left(\log n/\epsilon^2\hat{t}\right)$, then H is disconnected with at least a constant probability. So, it boils down to a connectivity check in H. The intuition is formalized in the following Lemma that can be proved using Markov's inequality.

Lemma 6. *Let G be a graph with $|V(G)| = n$, $\hat{t} \geq \frac{2000\log n}{\epsilon^2}|\textsc{Cut}(G)|$ and $\epsilon \in (0,1)$. If $H(V(G), E_p)$ be a subgraph of G where each edge in $E(G)$ is included in E_p independently with probability $p = \min\left\{\frac{200\log n}{\epsilon^2\hat{t}}, 1\right\}$, then H is connected with probability at most $\frac{1}{10}$.*

Before moving to the next section, we prove Lemma 5 and 6 here.

Proof of Lemma 5. If $p = 1$, we are done as the graph H is exactly same as that of G. So, without loss of generality assume that the graph G is connected. Otherwise, the lemma holds trivially as $|\textsc{Cut}(G)| = 0$, i.e., $\hat{t} = 0$ and $p = 1$. Hence, for the rest of the proof we will assume that $p = \frac{200 \log n}{\epsilon^2 \hat{t}}$.

Consider a cut $\mathcal{C}_G(A_1, A_2)$ of size k in G. As we are sampling each edge with probability p, the expected size of the cut $\mathcal{C}_H(A_1, A_2)$ is pk. Using Chernoff bound (see Lemma 17 in Section A), we get

$$\mathbb{P}\left(|\mathcal{C}_H(A_1, A_2) - pk| \geq \epsilon pk\right) \leq e^{-\epsilon^2 pk/3\hat{t}} = n^{-\frac{100k}{3\hat{t}}} \tag{1}$$

Note that here we want to show that every cut in G is approximately preserved in H. To do so, we will use Lemma 4 along with Equation 1 as follows. Let $Z_1, Z_2, \ldots, Z_\ell$ be the partition of the set of all cuts in G such that each cut in Z_j has the number of edges between $[j \cdot |\textsc{Cut}(G)|, (j+1)\,|\textsc{Cut}(G)|]$, where $\ell \leq \frac{n}{|\textsc{Cut}(G)|}$ and $j \leq \ell - 1$. From Lemma 4, $|Z_j| \leq n^{2j}$. Consider a particular $Z_j, j \in [\ell]$. Using the union bound along with Equation 1, the probability that there exists a cut in Z_j that is not approximately preserved in H is at most $\frac{1}{n^{11}}$. Taking union bound over all Z_j's, the probability that there exists a cut in G that is not approximately preserved is at most $\frac{1}{n^{10}}$. □

Proof of Lemma 6. Let $\mathcal{C}_G(A_1, A_2)$ be a minimum cut in G. Observe that

$$\mathbb{E}\left[|\mathcal{C}_H(A_1, A_2)|\right] = p\,|\mathcal{C}_G(A_1, A_2)| = p\,|\textsc{Cut}(G)|.$$

The result follows from Markov's inequality.

$$\mathbb{P}\left(G \text{ is connected}\right) \leq \mathbb{P}\left(|\mathcal{C}_G(A_1, A_2)| \geq 1\right) \leq \mathbb{E}[|\mathcal{C}_G(A_1, A_2)|] \leq \frac{1}{10}.$$

□

2.2 Formal Algorithm (Proof of Theorem 1)

In this Section, the main algorithm for minimum cut estimation is described in Algorithm 3 (Estimator) that makes multiple calls to Algorithm 2 (Verify-Guess). The Verify-Guess subroutine in turn calls Algorithm 1 (Sample) multiple times.

Given degree sequence of the graph G, that can be obtained using degree queries, we will first show how to independently sample each edge of G with probability p using only Neighbor queries.

Algorithm 1: Sample(D, p)

Input: $D = \{d(i) : i \in [n]\}$, where $d(i)$ denotes the degree of the i-th vertex in the graph G, and $p \in (0, 1]$.

Output: Return a subgraph $H(V, E_p)$ of $G(V, E)$ where each edge in $E(G)$ is included in E_p with probability p.

```
Set q = 1 − √(1 − p) and m = (Σ_{i=1}^n d_i)/2
for (each i ∈ [n]) do
    for (each j ∈ [d(i)] with d(i) > 0) do
        // Let r_j be the j-th neighbor of the i-th vertex
        Add the edge (i, r_j) to the set E_p with probability q;
    end
    Return the graph H(V, E_p).
end
```

The following lemma proves the correctness of the above algorithm Sample(D, p).

Lemma 7. *$\textsc{Sample}(D, p)$ returns a random subgraph $H(V(G), E_p)$ of G such that each edge $e \in E$ is included in E_p independently with probability p. Moreover, in expectation, the number of* Neighbor *queries made by $\textsc{Sample}(D, p)$ is at most $2pm$.*

Proof. From the description of $\textsc{Sample}(D, p)$, it is clear that the probability that a particular edge $e \in E(G)$ is added to E_p with probability $1 - (1 - q)^2 = p$.

Observe, $\mathbb{E}[|E_p|] = pm$. The bound on the number of Neighbor queries now follows from the fact that $\textsc{Sample}(D, p)$ makes at most $2|E_p|$ many Neighbor queries. □

One of the core ideas behind the proof of Theorem 1 is that, given an estimate $\hat{t}$ of t, we want to efficiently (in terms of number of local queries used by the algorithm) decide if $\hat{t} \leq t$ or if $\hat{t} \gtrsim \frac{\log n}{\epsilon^2} \times t$. Using Algorithm 2, we will show that this can be done using $\widetilde{\mathcal{O}}\left(m/\hat{t}\right)$ many Neighbor queries in expectation. Another interesting feature of Algorithm 2 is that, if estimate $\hat{t} \leq t$, then Algorithm 2 outputs an estimate which is a $(1 \pm \epsilon)$-approximation of t.

Algorithm 2: $\textsc{Verify-Guess}(D, \hat{t}, \epsilon)$

Input: $D = \{d(i) : i \in [n]\}$, where $d(i)$ denotes the degree of the i-th vertex in the graph G and $m = \frac{1}{2}\sum\limits_{i=1}^{n} d(i) \geq n - 1$. Also, a guess $\hat{t}$, with $1 \leq \hat{t} \leq \frac{n}{2}$, for the size of the global minimum cut in G, and $\epsilon \in (0, 1)$.

Output: The algorithm should "Accept" or "Reject" $\hat{t}$, with high probability, depending on the following

- If $\hat{t} \leq |\textsc{Cut}(G)|$, then Accept $\hat{t}$ and also output an $(1 \pm \epsilon)$-approximation of $|\textsc{Cut}(G)|$
- If $\hat{t} \geq \frac{200 \log n}{\epsilon^2} |\textsc{Cut}(G)|$, then Reject $\hat{t}$

Set $p = \min\left\{\frac{200 \log^2 n}{\epsilon^2 \hat{t}}, 1\right\}$.
Set $\Gamma = 100 \log n$ and Call $\textsc{Sample}(D, p)$ Γ times.
If at least one $\textsc{Sample}(D, p)$ returns Fail, then Return Fail as the output of
  Verify-Guess and Quit.
Otherwise, Let $H_i(V, E_p^i)$ be the output of i-th call to $\textsc{Sample}(D, p)$, where $i \in [\Gamma]$
if (*at least* $\Gamma/2$ *many* $H_i's$ *are disconnected*) **then**
  Reject $\hat{t}$
end
else if *(all H_i's are connected)* **then**
  Accept $\hat{t}$, find $\textsc{Cut}(H_i)$ for any $i \in [\Gamma]$, and return $\tilde{t} = \frac{|\textsc{Cut}(H_i)|}{p}$.
end
else
  Return Fail.
  // When we cannot decide between "Reject" or "Accept" it will return Fail
end

The following lemma proves the correctness of Algorithm 2. The lemmas used in proof are Lemmas 5, 6 and 7.

Lemma 8. *$\textsc{Verify-Guess}(D, \hat{t}, \epsilon)$ in expectation makes $\widetilde{\mathcal{O}}\left(\frac{m}{\hat{t}}\right)$ many* Neighbor *queries to the graph G and behaves as follows:*

(i) If $\hat{t} \geq \frac{2000 \log n}{\epsilon^2} |\textsc{Cut}(G)|$, then $\textsc{Verify-Guess}(D, \hat{t}, \epsilon)$ *rejects $\hat{t}$ with probability at least $1 - \frac{1}{n^9}$.*

(ii) If $\hat{t} \leq |\textsc{Cut}(G)|$, then $\textsc{Verify-Guess}(D, \hat{t}, \epsilon)$ *accepts $\hat{t}$ with probability at least $1 - \frac{1}{n^9}$. Moreover, in this case,* $\textsc{Verify-Guess}(D, \hat{t}, \epsilon)$ *reports an $(1 \pm \epsilon)$-approximation to* $\textsc{Cut}(G)$.

Proof. $\textsc{Verify-Guess}(D, \hat{t}, \epsilon)$ calls $\textsc{Sample}(D, \epsilon)$ for $\Gamma = 100 \log n$ times with p being set to $\min\left\{\frac{200 \log n}{\epsilon^2 \hat{t}}, 1\right\}$. Recall, from Lemma 7, that each call to $\textsc{Sample}(D, p)$ makes in expectation at most $2pm$ many $\textsc{Neighbor}$ queries, and returns a random subgraph $H(V, E_p)$, where each edge in $E(G)$ is included in E_p with probability p. So, $\textsc{Verify-Guess}(D, \hat{t}, \epsilon)$ makes in expectation $\mathcal{O}(pm \log n) = \widetilde{\mathcal{O}}\left(m/\hat{t}\right)$ many $\textsc{neighbor}$ queries and generates Γ many random subgraphs of G. The subgraphs are denoted by $H_1(V, E_p^1), \ldots, H_\Gamma(V, E_p^\Gamma)$.

(i) Let $\hat{t} \geq \frac{2000 \log n}{\epsilon^2} |\textsc{Cut}(G)|$. From Lemma 6, we have that H_i will be connected with probability at most $\frac{1}{10}$. Observe that in expectation, we get that at least $\frac{9\Gamma}{10}$ many H_i's will be disconnected. By Chernoff bound (see Lemma 17 in Section A), the probability that at most $\frac{\Gamma}{2}$ many H_i's are disconnected is at most $\frac{1}{n^{10}}$. Therefore, $\textsc{Verify-Guess}(D, \hat{t}, \epsilon)$ rejects any $\hat{t}$ satisfying $\hat{t} \geq \frac{2000 \log n}{\epsilon^2} |\textsc{Cut}(G)|$ with probability at least $1 - \frac{1}{n^9}$.

(ii) Let $\hat{t} \leq |\textsc{Cut}(G)|$. Using Lemma 5, we have that every cut of size k in G has size $pk(1 \pm \epsilon)$ in H_i with probability at least $1 - \frac{1}{n^{10}}$. Therefore, with probability at least $1 - \frac{\Gamma}{n^{10}}$, for all $i \in [\Gamma]$, every cut of size k in G has size $pk(1 \pm \epsilon)$ in H_i. This implies that if $\hat{t} \leq |\textsc{Cut}(G)|$ then $\textsc{Verify-Guess}(D, \hat{t}, \epsilon)$ accepts any $\hat{t}$ with probability at least $1 - \frac{1}{n^9}$. Moreover, for any H_i, observe that $\frac{|\textsc{Cut}(H_i)|}{p}$ is an $(1 \pm \epsilon)$-approximation to $|\textsc{Cut}(G)|$. Hence, when $\hat{t} \leq |\textsc{Cut}(G)|$, $\textsc{Verify-Guess}(D, \hat{t}, \epsilon)$ also returns an $(1 \pm \epsilon)$ approximation to $|\textsc{Cut}(G)|$ with probability $1 - \frac{1}{n^9}$.

□

$\textsc{Estimator}(\epsilon)$ (Algorithm 3) will estimate the size of the minimum cut in G using $\textsc{Degree}$ and $\textsc{Neighbor}$ queries. The main subroutine used by the algorithm will be $\textsc{Verify-Guess}(D, \hat{t}, \epsilon)$.

The following lemma shows that with high probability $\textsc{Estimator}(\epsilon)$ correctly estimates the size of the minimum cut in the graph G, and it also bounds the expected number of queries used by the algorithm.

Lemma 9. $\textsc{Estimator}$*(ϵ) returns $(1 \pm \epsilon)$ approximation to $|\textsc{Cut}(G)|$ with probability at least $1 - \frac{1}{n^8}$ by making in expectation $\min\left\{m + n, \frac{m}{t}\right\}$ poly$\left(\log n, \frac{1}{\epsilon}\right)$ many queries and each query is either a* $\textsc{Degree}$ *or a* $\textsc{Neighbor}$ *query to the unknown graph G.*

Proof. Without loss of generality, assume that n is a power of 2. If $m < n - 1$ or if there exists a $i \in [n]$ such that $d_i = 0$ then the graph G is disconnected. In this case the algorithm $\textsc{Estimator}(\epsilon)$ makes n $\textsc{Degree}$ queries and returns the correct answer. Thus we assume that $m \geq n - 1$.

First, we prove the correctness and query complexity when the graph is connected, that is, $t \geq 1$. Note that $\textsc{Estimator}(\epsilon)$ calls $\textsc{Verify-Guess}(D, \hat{t}, \epsilon)$ for different values of $\hat{t}$ starting from n. Recall that $\kappa = \frac{2000 \log n}{\epsilon^2}$. For a particular $\hat{t}$ with $\hat{t} \geq \kappa t$, $\textsc{Verify-Guess}(D, \hat{t}, \epsilon)$ accepts $\hat{t}$ with probability at most $\frac{1}{n^9}$ by Lemma 8 (i). So, by the union bound, the probability that $\textsc{Verify-Guess}(D, \hat{t}, \epsilon)$ accepts some $\hat{t}$ with $\hat{t} \geq \kappa t$, is at most $\frac{\log n}{n^9}$. Hence, with probability at least $1 - \frac{\log n}{n^9}$, we can say that the $\textsc{Verify-Guess}(D, \hat{t}, \epsilon)$ rejects all $\hat{t}$ with $\hat{t} \geq \kappa t$.

Algorithm 3: ESTIMATOR(ϵ)

Input: DEGREE and NEIGHBOR query access to an unknown graph G, and a parameter $\epsilon \in (0, 1)$.

Output: An $(1 \pm \epsilon)$-approximation to $t = |\text{CUT}(G)|$.

Find the degrees of all the vertices in G by making n many DEGREE queries. Let $D = \{d(1), \ldots, d(n)\}$, where $d(i)$ denotes the degree of the i-th vertex in G.

If $\exists i \in [n]$ such that $d(i) = 0$, then return $t = 0$ and QUIT. Otherwise, proceeds as follows.

Find $m = \frac{1}{2} \sum\limits_{i=1}^{n} d(i)$. If $m < n - 1$, return $t = 0$ and QUIT. Otherwise, proceed as follows.

Set $\kappa = \frac{2000 \log n}{\epsilon^2}$

Initialize $\hat{t} = \frac{n}{2}$.

while *($\hat{t} \geq 1$)* **do**

  Call VERIFY-GUESS($D, \hat{t}, \epsilon$).

  if *(*VERIFY-GUESS($D, \hat{t}, \epsilon$) *returns* FAIL*)* **then**

    Return FAIL as the output of ESTIMATOR(ϵ).

  end

  else if *(*VERIFY-GUESS($D, \hat{t}, \epsilon$) *rejects* $\hat{t}$*)* **then**

    set $\hat{t} = \frac{\hat{t}}{2}$ and continue.

  end

  else

    Set $\hat{t}_u = \max\left\{\frac{\hat{t}}{\kappa}, 1\right\}$.

    Call VERIFY-GUESS($D, \hat{t}_u, \epsilon$).

    If VERIFY-GUESS returns FAIL or reject $\hat{t}_u$, then return FAIL as the output of ESTIMATOR(ϵ). Otherwise, let $\tilde{t}$ be the output of VERIFY-GUESS.

    Return $\tilde{t}$ as the output of ESTIMATOR(ϵ)

  end

end

Output: Return that the graph G is disconnected.

Observe that, from Lemma 8 (ii), the first time $\hat{t}$ satisfy the following inequality

$$\frac{t}{2} < \hat{t} \leq t,$$

VERIFY-GUESS($D, \hat{t}, \epsilon$) will accept $\hat{t}$ with probability at least $1 - \frac{1}{n^9}$. Therefore, ESTIMATOR(ϵ) accepts a $\hat{t}$ with $\frac{t}{2} < \hat{t} < \kappa t$ with probability $1 - \frac{\log n + 1}{n^9}$. From the description of ESTIMATOR(ϵ), note that, we find $\hat{t}_u$ by dividing $\hat{t}$ by κ. Now that we have $\hat{t_u} < t$, we will call procedure VERIFY-GUESS($D, \hat{t}, \epsilon$) with $\hat{t} = \hat{t}_u$. By Lemma 8 (ii), VERIFY-GUESS($D, \hat{t}, \epsilon$) accepts and reports an $(1 \pm \epsilon)$ approximation to t with probability at least $1 - \frac{1}{n^9}$.

We will now analyze the number of DEGREE and NEIGHBOR queries made by the algorithm. We make an initial n many queries to construct the set D. Then at the worst case, we call VERIFY-GUESS($D, \hat{t}, \epsilon$) for $\hat{t} = \frac{n}{2}, \ldots, t'$ and $\hat{t} = \frac{t'}{\kappa} \geq \frac{t}{2\kappa}$, where $\frac{t}{2} < t' < \kappa t$. It is because VERIFY-GUESS($D, \hat{t}, \epsilon$) accepts $\hat{t}$ with probability $1 - \frac{1}{n^9}$ when the first time $\hat{t}$ satisfy the inequality $\hat{t} \leq t$. Hence, by Lemma 8 and the facts that $n \leq \frac{m}{t}$ and $\hat{t}_u \geq \frac{t}{2\kappa}$ with probability at least $1 - \frac{\log n + 1}{n^9}$, in expectation the total number of queries made by the algorithm is at most

$$n + \log n \cdot \left(1 - \frac{\log n + 1}{n^9}\right) \cdot \widetilde{\mathcal{O}}\left(\frac{2\kappa m}{t}\right) + \log n \cdot \left(\frac{\log n + 1}{n^9}\right) \cdot \widetilde{\mathcal{O}}(m) = \widetilde{\mathcal{O}}\left(\frac{m}{t}\right).$$

Note that each query made by $\textsc{Estimator}(\epsilon)$ is either a $\textsc{Degree}$ or a $\textsc{Neighbor}$ query.

Now we analyze the case when $t = 0$. Observe that $\textsc{Verify-Guess}(D, \hat{t}, \epsilon)$ rejects all $\hat{t} \geq 1$ with probability $1 - \frac{\log n}{n^9}$, and therefore, $\textsc{Estimator}(\epsilon)$ will report $t = 0$. As we have called $\textsc{Verify-Guess}(D, \hat{t}, \epsilon)$ for all $\hat{t} = \frac{n}{2}, \ldots, 1$, the number of queries made by $\textsc{Estimator}(\epsilon)$, in the case when $t = 0$, is $\widetilde{\mathcal{O}}(m) + n$. Note that the additional n term in the bound comes from the fact that to compute D the algorithms needs to make n many $\textsc{Degree}$ queries. $\square$

3 Lower bounds

In this Section, we prove Theorems 2 and 3 using reductions from suitable problems in communication complexity. In Section 3.1, we discuss about two party communication complexity along with the problems that will be used in our reductions. We will discuss the proofs of Theorems 2 and 3 in Section 3.2.

3.1 Communication Complexity

In two-party communication complexity there are two parties, Alice and Bob, that wish to compute a function $\Pi : \{0,1\}^N \times \{0,1\}^N \to \{0,1\} \cup \{0,1\}^n$ [2]. Alice is given $\mathbf{x} \in \{0,1\}^N$ and Bob is given $\mathbf{y} \in \{0,1\}^N$. Let x_i (y_i) denotes the i-th bit of $\mathbf{x}$ ($\mathbf{y}$). While the parties know the function Π, Alice does not know $\mathbf{y}$, and similarly Bob does not know $\mathbf{x}$. Thus they communicate bits following a pre-decided protocol $\mathcal{P}$ in order to compute $\Pi(\mathbf{x}, \mathbf{y})$. We say a randomized protocol $\mathcal{P}$ computes Π if for all $(\mathbf{x}, \mathbf{y}) \in \{0,1\}^N \times \{0,1\}^N$ we have $\mathbb{P}[\mathcal{P}(\mathbf{x}, \mathbf{y}) = \Pi(\mathbf{x}, \mathbf{y})] \geq 2/3$. The model provides the parties access to common random string of arbitrary length. The cost of the protocol $\mathcal{P}$ is the maximum number of bits communicated, where maximum is over all inputs $(\mathbf{x}, \mathbf{y}) \in \{0,1\}^N \times \{0,1\}^N$. The communication complexity of the function is the cost of the most efficient protocol computing Π. For more details on communication complexity see [Kus97]. We now define two functions k-$\textsc{Intersection}$ and $\textsc{Find}$-k-$\textsc{Intersection}$ and discuss their communication complexity. Both these functions will be used in our reductions.

Definition 10 ($\textsc{Find}$-k-$\textsc{Intersection}$)**.** Let $k, N \in \mathbb{N}$ such that $k \leq N$. Let $S = \{(\mathbf{x}, \mathbf{y}) \in \{0,1\}^N \times \{0,1\}^N : \sum_{i=1}^{N} x_i y_i = k\}$. The $\textsc{Find}$-k-$\textsc{Intersection}$ function on N bits is a partial function and is defined as $\text{FIND-INT}_k^N : S \to \{0,1\}^N$, and is defined as

$$\text{FIND-INT}_k^N(\mathbf{x}, \mathbf{y}) = \mathbf{z}, \text{where } z_i = x_i y_i \text{ for each } i \in [N].$$

Note that the objective is that at the end of the protocol Alice and Bob know $\mathbf{z}$.

Definition 11 (k-$\textsc{Intersection}$)**.** Let $k, N \in \mathbb{N}$ such that $k \leq N$. Let $S = \{(\mathbf{x}, \mathbf{y}) : \sum\limits_{i=1}^{N} x_i y_i = k \text{ or } k-1\}$. The k-$\textsc{Intersection}$ function on N bits is a partial function denoted by $\text{INT}_k^N : S \to \{0,1\}$, and is defined as follows:

$$\text{INT}_k^N(\mathbf{x}, \mathbf{y}) = \begin{cases} 1 & \text{if } \sum_{i=1}^{N} x_i y_i = k \\ 0 & \text{otherwise} \end{cases}$$

[2] The co-domain of Π looks odd, as the the co-domain is $\{0,1\}$ usually. However, we need $\{0,1\} \cup \{0,1\}^n$ to take care of all the problems in communication complexity we discuss in this paper

In communication complexity, the k-INTERSECTION function on N bits when $k = 1$ is known as DISJOINTNESS function on N.

Lemma 12. *Let $k, N \in \mathbb{N}$ such that $k \leq cN$ for some constant $c < 1$. The randomized communication complexity of* FIND-k-INTERSECTION *function on N bits is $\Omega(N)$.*

Lemma 13. *Let $k, N \in \mathbb{N}$ such that $k \leq cN$ for some constant $c < 1$. The randomized communication complexity of* k-INTERSECTION *function on N bits (INT_k^N) is $\Omega(N)$.*

3.2 Proofs of Theorems 2 and 3

The proofs of Theorems 2 and 3 are inspired from the lower bound proof of Eden and Rosenbaum [ER18] for estimating MINCUT [3].

Proof of Theorem 2. We prove by giving a reduction from FIND-$t/2$-INTERSECTION on N bits. Without loss of generality assume that t is even. Let $\mathbf{x}$ and $\mathbf{y}$ be the inputs of Alice and Bob. Note that $\sum_{i=1}^{N} x_i y_i = t/2$.

We first discuss a graph $G_{(\mathbf{xy})}(V, E)$ that can be generated from $(\mathbf{x}, \mathbf{y})$, such that $|V| = n$ and $|E| = m \geq 2nt$, and works as the 'hard' instance for our proof. Note that $G_{(\mathbf{x},\mathbf{y})}$ should be such that no useful information about the MINCUT can be derived by knowing only one of $\mathbf{x}$ and $\mathbf{y}$. Let $s = t + \sqrt{t^2 + (m - nt)/2}$ and $N = s^2$. In particular, $2t \leq s \leq 2t + 3\sqrt{m}$. Also, $s \geq \sqrt{m/2}$ and therefore $s = \Theta(\sqrt{m})$.

The graph $G_{(\mathbf{x},\mathbf{y})}$ and its properties:

$G_{(\mathbf{x},\mathbf{y})}$ has the following structure.

- $V = S_A \cup T_A \cup S_B \cup T_B \cup C$ such that $|S_A| = |T_A| = |S_B| = |T_B| = s$ and $|C| = n - 4s$. Let $S_A = \{s_i^A : i \in [s]\}$ and similarly $T_A = \{t_i^A : i \in [s]\}$, $S_B = \{s_i^B : i \in [s]\}$ and $T_B = \{t_i^B : i \in [s]\}$.
- Each vertex in C is connected to $2t$ different vertices in S_A.
- For $i, j \in [s]$: if $x_{ij} = y_{ij} = 1$, then $(s_i^A, t_j^B) \in E$ and $(s_i^B, t_j^A) \in E$; otherwise, $(s_i^A, t_j^A) \in E$ and $(s_i^B, t_j^B) \in E$.

Observation 14. $G_{(\mathbf{x},\mathbf{y})}$ satisfies the following properties.

Property-1: The degree of every vertex in C is $2t$. For any $v \notin C$, the neighbors of v inside C are fixed irrespective of $\mathbf{x}$ and $\mathbf{y}$; and the number of neighbors outside C is $s \geq 2t$.

Property-2: There are t edges between the vertex sets $(C \cup S_A \cup T_A)$ and $(S_B \cup T_B)$, and removing them $G_{(\mathbf{x},\mathbf{y})}$ becomes disconnected.

Property-3: Every pair of vertices $(S_A \cup T_A \cup C)$ is connected by at least $3t/2$ edge disjoint paths. Also, every pair of vertices in $(S_B \cup T_B)$ is connected by at least $3t/2$ edge disjoint paths.

Property-4: The set of t edges between the vertex sets $(C \cup S_A \cup T_A)$ and $(S_B \cup T_B)$ forms the unique global minimum cut of $G(\mathbf{x}, \mathbf{y})$,

[3] Note that Eden and Rosenbaum [ER18] stated the result in terms k-Edge Connectivity.

Property-5: $x_{ij} = y_{ij} = 1$ if and only if (s_i^A, t_j^B) and (s_i^B, t_j^A) are the edges in the unique global minimum cut of $G_{(\mathbf{x},\mathbf{y})}$.

Proof. Property-1 and Property-2 directly follow from the construction. Now, we will prove Property-3. We first show that every pair of vertices $(S_A \cup T_A \cup C)$ is connected by at least $3t/2$ edge disjoint paths by breaking the analysis into the following cases.

(i) Consider $s_i^A, s_j^A \in S_A$, for $i, j \in [s]$. Under the promise that $\sum_{i=1}^{N} x_i y_i = t/2$, s_i^A, s_j^A have at least $s - t \geq 3t/2$ common neighbors in T_A and thus there are at least $3t/2$ edge disjoint paths connecting them.

(ii) Consider $s_i^A \in S_A$ and $t_j^A \in T_A$, for $i, j \in [s]$. Let $s_{j_1}^A, \ldots, s_{j_{3t/2}}^A$ be $3t/2$ distinct neighbors of t_j^A in S_A. Since, s_i^A has $3t/2$ common neighbors with each $s_{j_r}^A$, $r \in [3t/2]$, there is a matching of size $3t/2$. Denote this matching by $(t_{j_r}^A, s_{j_r}^A)$, $r \in [3t/2]$. Thus $(s_i^A, t_{j_r}^A), (t_{j_r}^A, s_{j_r}^A), (s_{j_r}^A, t_j^A)$, for $r \in [3t/2]$, forms a set of edge disjoint paths of size $3t/2$ from s_i^A to t_j^A, each of length 3. In case s_i^A is one of the neighbors of t_j^A, then one of the $3t/2$ paths gets reduced to (s_i^A, t_j^A), a length 1 path that is edge disjoint from the remaining paths.

(iii) Consider $u, v \in C$. Let $u_1, \ldots, u_{2t} \in S_A$ and $v_1, \ldots, v_{2t} \in S_A$ be the neighbors of u and v respectively in S_A. If for some $i, j \in [2t]$, $u_i = v_j$ then $(u, u_i), (u_i, v_j), (v_j, v)$ is a desired path. Thus, assume $u_i \neq v_j$ for all $i, j \in [2t]$. For all $i \in [2t]$, since u_i and v_i have at least $3t/2$ common neighbors in T_A we can find $3t/2$ edge disjoint paths $(u_i, t_i^A), (t_i^A, v_i)$, where $t_i^A \in T^A$. Existence of $3t/2$ edge disjoint paths from $u \in C$ to $v \in S_A$ can be proved as in (i). and from $u \in C$ to $v \in T_A$ can be proved as in (ii).

Similarly, we can show that every pair of vertices in $(S_B \cup T_B)$ is connected by $3t/2$ many edge disjoint paths.

Observe that Property-4 follows from Property-3, and Property-5 follows from the construction of $G_{(\mathbf{x},\mathbf{y})}$ and Property-4. □

Now, by contradiction assume that there exists an algorithm $\mathcal{A}$ that makes $o(m)$ queries to $G_{(\mathbf{x},\mathbf{y})}$ and finds all the edges of a global minimum cut with probability $2/3$. Now, we give a protocol $\mathcal{P}$ for Find-$t/2$-Intersection on N bits when the $\mathbf{x}$ and $\mathbf{y}$ are the inputs of Alice and Bob, respectively. Note that $x, y \in \{0,1\}^N$ such that $\sum_{i=1}^{N} x_i y_i = t/2$.

Protocol $\mathcal{P}$ for Find-$t/2$-Intersection**:**

Alice and Bob run the query algorithm $\mathcal{A}$ when the unknown graph is $G_{(\mathbf{x},\mathbf{y})}$. Now we explain how they simulate the local queries and random edge query on $G_{(\mathbf{x},\mathbf{y})}$ by communication. We would like to note that each query can be answered deterministically.

Degree **query:** By Property-1, the degree of every vertex does not depend on the inputs of Alice and Bob, and therefore any degree query can be simulated without any communication.

Neighbor **query:** For $v \in C$, the set of $2t$ neighbors are fixed by the construction. So, any neighbor query involving any $v \in C$ can be answered without any communication. For $i \in [s]$ and $s_i^A \in S_A$, let $N_C(s_i^A)$ be the set of fixed neighbors of s_i^A inside C. So, by Property-1, $d(s_i^A) = \left|N_C(s_i^A)\right| + s$ [4]. The labels of the neighbors of s_i^A are such that the first $\left|N_C(s_i^A)\right|$ many neighbors are inside C, and they are arranged in a fixed but arbitrary order. For $j \in [s]$,

[4] $d(u)$ denotes the degree of the vertex u in $G_{(\mathbf{x},\mathbf{y})}$

the $(|N_C(v)| + j)$-th neighbor of s_i^A is either t_j^B or s_j^A depending on whether $x_{ij} = y_{ij} = 1$ or not, respectively. So, any neighbor query involving vertex in S_A can be answered by 2 bits of communication. Similar arguments also hold for the vertices in $S_B \cup T_A \cup T_B$.

ADJACENCY **query:** Observe that each adjacency query can be answered by at most 2 bits of communication, and it can be argued like the NEIGHBOR query.

RANDOM EDGE **query:** By Property-1, the degree of any vertex $v \in V$ is independent of the inputs of Alice and Bob. Alice and Bob use shared randomness to sample a vertex in V proportional to its degree. Let $r \in V$ be the sampled vertex. They again use shared randomness to sample an integer j in $[d(v)]$ uniformly at random. Then they determine the j-th neighbor of r using NEIGHBOR query. Observe that this procedure simulates a RANDOM EDGE query by using at most 2 bits of communication.

Using the fact that $G_{(\mathbf{x},\mathbf{y})}$ satisfies Property-4 and 5, the output of algorithm $\mathcal{A}$ determines the output of protocol $\mathcal{P}$ for FIND-$t/2$-INTERSECTION. As each query of $\mathcal{A}$ can be simulated by at most two bits of communication by the protocol $\mathcal{P}$, the number of bits communicated is $o(m)$. Recall that $N = s^2$ and $s = \Theta(\sqrt{m})$. So, the number of bits communicated by Alice and Bob in $\mathcal{P}$ is $o(N)$. This contradicts Theorem 12. □

Proof of Theorem 3. The proof of this theorem uses the same construction as the one used in the proof of Theorem 2. The 'hard' communication problem to reduce from is $t/2$-INTERSECTION (see Definition 11) on N bits, where $N = s^2$ and $s = \Theta(\sqrt{m})$. □

4 Conclusion

Global minimum r-way cut. Global minimum r-cut, for a graph $G = ([n], E)$, $|V| = n$ and $|E| = m$, is a partition of the vertex set $[n]$ into r-sets $S_1, \ldots, S_r$ such that the following is minimized

$$|\{\{i,j\} \in E \ : \ \exists k, \ell \, (k \neq \ell) \in [r], \text{ with } i \in S_k \text{ and } j \in S_\ell\}| \, .$$

Let $\text{CUT}_r(G)$ denote the set of edges corresponding to a minimum r-cut, i.e., the edges that goes across different partitions, and by the size of minimum r-cut, we mean $|\text{CUT}_r(G)|$. The sampling and verification idea used in the proof of Theorem 1 can be extended directly, together with [Kar93, Corollary 8.2], to get the following result.

Theorem 15. *There exists an algorithm, with* DEGREE *and* NEIGHBOR *query access to an unknown graph $G = ([n], E)$, that with high probability outputs a $(1 \pm \epsilon)$-approximation of the size of the minimum r-cut of G. The expected number of queries used by the algorithm is*

$$\min\left\{m + n, \frac{m}{t_r}\right\} poly\left(r, \log n, \frac{1}{\epsilon}\right),$$

where $t_r = |\text{CUT}_r(G)|$.

Minimum cuts in simple multigraphs. A graph with multiple edges between a pair of vertices in the graph but without any self loops are called *simple multigraphs.* If we have DEGREE and NEIGHBOR[5] query access to simple multigraphs then we can directly get the following generalization of Theorem 1.

[5]For simple multigraphs, we will assume that the neighbors of a vertex are stored with multiplicities.

Theorem 16. (Minimum cut estimation in simple multigraphs using local queries) *There exists an algorithm, with* Degree *and* Neighbor *query access to an unknown simple multigraph $G = (V, E)$, that solves the minimum cut estimation problem with high probability. The expected number of queries used by the algorithm is*

$$\min\left\{m + n, \frac{m}{t}\right\} poly\left(\log n, \frac{1}{\epsilon}\right),$$

where n is the number of vertices in the multigraph, m is the number of edges in the multigraph and t is the number of edges in a minimum cut.

References

[ABG+18] Maryam Aliakbarpour, Amartya Shankha Biswas, Themis Gouleakis, John Peebles, Ronitt Rubinfeld, and Anak Yodpinyanee. Sublinear-Time Algorithms for Counting Star Subgraphs via Edge Sampling. *Algorithmica*, 80(2):668–697, 2018.

[AGM12] Kook Jin Ahn, Sudipto Guha, and Andrew McGregor. Graph Sketches: Sparsification, Spanners, and Subgraphs. In *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS*, pages 5–14, 2012.

[AKK19] Sepehr Assadi, Michael Kapralov, and Sanjeev Khanna. A Simple Sublinear-Time Algorithm for Counting Arbitrary Subgraphs via Edge Sampling. In *ITCS*, pages 6:1–6:20, 2019.

[BBM11] Eric Blais, Joshua Brody, and Kevin Matulef. Property Testing Lower Bounds via Communication Complexity. In *Proceedings of the 26th Annual IEEE Conference on Computational Complexity, CCC*, pages 210–220, 2011.

[DP09] Devdatt P. Dubhashi and Alessandro Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms.* Cambridge University Press, 1st edition, 2009.

[ELRS17] T. Eden, A. Levi, D. Ron, and C. Seshadhri. Approximately Counting Triangles in Sublinear Time. *SIAM J. Comput.*, 46(5):1603–1646, 2017.

[ER18] Talya Eden and Will Rosenbaum. Lower Bounds for Approximating Graph Parameters via Communication Complexity. In *Proceedings of the 21st International Conference on Approximation Algorithms for Combinatorial Optimization Problems, APPROX*, pages 11:1–11:18, 2018.

[ERS18] Talya Eden, Dana Ron, and C. Seshadhri. On Approximating the Number of k-Cliques in Sublinear Time. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 722–734, 2018.

[Fei06] Uriel Feige. On sums of independent random variables with unbounded variance and estimating the average degree in a graph. *SIAM J. Comput.*, 35(4):964–984, 2006.

[GGR98] Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998.

[Gol17] Oded Goldreich. *Introduction to Property Testing.* Cambridge University Press, 2017.

[GPRW20] Andrei Graur, Tristan Pollner, Vidhya Ramaswamy, and S. Matthew Weinberg. New Query Lower Bounds for Submodular Function Minimization. In Thomas Vidick, editor, *Proceedings of the 11th Innovations in Theoretical Computer Science Conference, ITCS*, volume 151, pages 64:1–64:16, 2020.

[GR08] O. Goldreich and D. Ron. Approximating Average Parameters of Graphs. *Random Structures & Algorithms*, 32(4):473–493, 2008.

[GRS11] M. Gonen, D. Ron, and Y. Shavitt. Counting Stars and Other Small Subgraphs in Sublinear-Time. *SIAM Journal on Discrete Mathematics*, 25(3):1365–1411, 2011.

[Kar93] David R. Karger. Global Min-cuts in RNC, and Other Ramifications of a Simple Min-Cut Algorithm. In *Proceedings of the 4th Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms, SODA*, pages 21–30, 1993.

[KKR04] Tali Kaufman, Michael Krivelevich, and Dana Ron. Tight bounds for testing bipartiteness in general graphs. *SIAM J. Comput.*, 33(6):1441–1483, 2004.

[KS93] David R. Karger and Clifford Stein. An $\widetilde{\mathcal{O}}\left(n^2\right)$ Algorithm for Minimum Cuts. In S. Rao Kosaraju, David S. Johnson, and Alok Aggarwal, editors, *Proceedings of the 25th Annual ACM Symposium on Theory of Computing, STOC*, pages 757–765, 1993.

[KT19] Ken-ichi Kawarabayashi and Mikkel Thorup. Deterministic edge connectivity in near-linear time. *J. ACM*, 66(1):4:1–4:50, 2019.

[Kus97] Eyal Kushilevitz. Communication complexity. In *Advances in Computers*, volume 44, pages 331–360. Elsevier, 1997.

[McG14] Andrew McGregor. Graph Stream Algorithms: A Survey. *SIGMOD Rec.*, 43(1):9–20, 2014.

[MN20] Sagnik Mukhopadhyay and Danupon Nanongkai. Weighted Min-Cut: Sequential, Cut-Query, and Streaming Algorithms. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC*, pages 496–509, 2020.

[RSW18] A. Rubinstein, T. Schramm, and S. M. Weinberg. Computing Exact Minimum Cuts Without Knowing the Graph. In *ITCS*, pages 39:1–39:16, 2018.

A Probability Results

Lemma 17 (See [DP09]). *Let $X = \sum_{i\in[n]} X_i$ where X_i, $i \in [n]$, are independent random variables, $X_i \in [0,1]$ and $\mathbb{E}[X]$ is the expected value of X. Then*

(i) For $\epsilon > 0$

$$\Pr[|X - \mathbb{E}[X]| > \epsilon\mathbb{E}[X]] \leq \exp\left(-\frac{\epsilon^2}{3}\mathbb{E}[X]\right).$$

(ii) Suppose $\mu_L \leq \mathbb{E}[X] \leq \mu_H$, then for $0 < \epsilon < 1$

(a) $\Pr[X > (1+\epsilon)\mu_H] \leq \exp\left(-\frac{\epsilon^2}{3}\mu_H\right)$.

(b) $\Pr[X < (1-\epsilon)\mu_L] \leq \exp\left(-\frac{\epsilon^2}{2}\mu_L\right)$.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il