A Simple Semi-Streaming Algorithm for Global Minimum Cuts

Sepehr Assadi* Aditi Dudeja†

Abstract

Recently, Rubinstein, Schramm, and Weinberg [ITCS'18] gave an algorithm for finding an exact global minimum cut of undirected graphs in the cut-query model in which the access to the graph is via querying the number of edges crossing a given cut. It was subsequently observed in the literature that this algorithm also implies that the minimum cut problem in the streaming model admits an $\widetilde{O}(n)$-space algorithm in only two passes over the input.

In this paper, we present a simpler and self-contained proof of this result in the streaming model with an improved space complexity that we show is within a sub-logarithmic factor of being optimal.

1 Introduction

Given an undirected graph $G = (V, E)$, the minimum cut problem asks for finding a set $S \subset V$ of vertices, namely, a *cut*, with a minimum number of crossing edges between S and $V \setminus S$. The minimum cut problem is a classical graph optimization problems with a wide range of applications; see, e.g., [15, 16, 19, 23–27] and references therein. We study this problem in the *semi-streaming* model [13]. A semi-streaming algorithm is given the input edges in an arbitrarily ordered stream and is allowed to make one or a few passes over this stream, while using a limited memory of $\widetilde{O}(n) := O(n \cdot \text{polylog}(n))$ bits; here, and throughout the paper, n denotes the number of vertices. Semi-streaming algorithms with a small number of passes are particularly successful in maintaining I/O-efficiency for processing massive graphs and as such have been studied extensively (see [30]).

It has been known for over a decade how to find a $(1+\varepsilon)$-approximate minimum cut via semi-streaming algorithms in one pass [2]. This is done by maintaining a *cut sparsifier* of the graph—a subgraph with $\widetilde{O}(n/\varepsilon^2)$ edges that preserves values of all cuts within a $(1 \pm \varepsilon)$-approximation [24]—and then returning a minimum cut of this sparsifier. It was also known that any single-pass algorithm for finding an exact minimum cut requires $\Omega(n^2)$ space [35]. Recently, [32] developed an algorithm in the *cut-query* model that finds a minimum cut by making $\widetilde{O}(n)$ queries that return a value of a given cut. While this was not originally a semi-streaming algorithm, it was observed in [7] that using streaming cut sparsifiers such as [2, 22], this approach leads to a surprising result in the streaming model: A semi-streaming algorithm for exact minimum cuts in *only two passes*! (This result was subsequently extended to weighted graphs with $O(\log n)$ passes [31]).

Our Results. The goal of this paper is to present a simpler and self-contained proof of this fundamental result for minimum cut directly in the semi-streaming model.

Theorem 1. *There is a semi-streaming algorithm that with high constant probability outputs an exact minimum cut of a given n-vertex graph in two passes and space of $O(n \log n)$ bits.*

*(sepehr.assadi@rutgers.edu) Department of Computer Science, Rutgers University.
†(aditi.dudeja@rutgers.edu) Department of Computer Science, Rutgers University.

Our proof of Theorem 1 is different from [32] and instead follows the recent edge-out contraction framework of [15] for the minimum cut problem. To obtain the precise bounds in Theorem 1 on the space and (more importantly) pass-complexity, several modifications to [15] are needed. As the goal of this paper is to present a complete and self-contained proof of a semi-streaming algorithm for minimum cut, we present a complete proof of Theorem 1 without relying on results of [15] but will point out how some of our intermediate lemmas relate to [15].

Theorem 1, besides simplifying and streamlining the result of [32] for semi-streaming algorithms, also improves the space complexity from $O(n \cdot \text{polylog}(n))$ for some unspecified $\text{polylog}(n)$-factor to only $O(n \log n)$ bits. This is already within a logarithmic factor of being optimal as $\Omega(n)$ bits are clearly necessary just to specify the output cut. We prove a new lower bound that implies that the space complexity of this algorithm is in fact even within a *sub-logarithmic* factor of optimal—the pass-complexity is also optimal in light of the $\Omega(n^2)$ lower bound of [35] for single-pass algorithms.

Theorem 2. *Any streaming algorithm that outputs the exact minimum cut value of given n-vertex graphs with probability more than half in constant $p > 1$ passes requires $\Omega(n \cdot (\log(n))^{1/2p-1})$ space.*

We conclude this section with the following remark which was an important motivation behind this paper in providing a self-contained and "streaming-friendly" proof of the result of [32].

Perspective: Beyond Iterative Multi-Pass Algorithms. Most of the *multi-pass* graph streaming algorithms in the literature for optimization problems are *iterative* algorithms. They maintain a partial solution and iteratively refine it during each pass, while making a "steady progress" toward the final answer. This is typically obtained by simulating a greedy algorithm in a pass-efficient manner via a Luby-style argument (see, e.g. [1,5,14,28,29]) or by running an iterative optimization method such as multiplicative weight update or gradient descent (see, e.g. [3,4,9,20]).

The algorithm in Theorem 1 is however based on a different principle: use the first pass to *sparsify* the graph while keeping one optimal solution intact, and then recover this sparsifier in the second pass and solve the problem offline. This approach seems to exploit the power of streaming algorithms in processing *sparse* graphs better than the iterative methods and lead to streaming algorithms with surprisingly smaller number of passes. We believe that this viewpoint is an important (non-technical) contribution of our work as it may pave the way for obtaining more pass-efficient streaming algorithms for other graph problems as well.

2 A Semi-Streaming Algorithm for Minimum Cut

We prove Theorem 1 in this section. The algorithm uses two passes and in each pass it performs a different task as follows: (*i*) The goal of the first pass is to contract some edges of the graph, without "destroying" at least one minimum cut, to reduce the number of vertices to $O(n/d_{\mathsf{min}})$ where d_{min} is the minimum degree of the graph (thus an upper bound on the minimum cut value); (*ii*) the goal of the second pass is to find a subgraph of this new graph with $O(n)$ edges which has the same minimum cut as in the original graph—at this point, the problem can be solved easily by storing all these $O(n)$ edges and finding a minimum cut of this final subgraph using any offline algorithm. We note that this high level approach is the same as the edge-out contraction framework of [15] and the key differences are in the way how each step is implemented.

We now formalize this. The main algorithm is simply as follows (to obtain the final algorithm, we simply repeat this algorithm $O(1)$ time in parallel to boost its success probability):

Algorithm 1. A two-pass streaming algorithm for the minimum cut problem.

- **First pass:** For every vertex $v \in V$, sample 2 edges incident on v independently and uniformly at random with repetition and store them during the stream. Let $G^{(2)}$ be the resulting graph and $V_1, \ldots, V_t$ be its connected components.

  In parallel, compute the minimum degree d_{min} of G and if $t > 100 \cdot n/d_{\mathsf{min}}$, terminate the algorithm and output FAIL.

- **Second pass:** Consider the multi-graph H obtained from G by contracting vertices in each set V_i into a single vertex and removing the self-loops.

  Let $F_1, \ldots, F_{d_{\mathsf{min}}}$ be initially empty. For each arriving edge e of H in the stream, include e in F_i where i is the smallest index such that $\{e\} \cup F_i$ contains no cycle as a subgraph of H; ignore e if no such i exists.

  At the end of the second pass, compute a minimum cut of the graph $F := F_1 \cup \ldots \cup F_{d_{\mathsf{min}}}$; if it contains less that d_{min} edges return this cut (after expanding the contracted vertices) as the minimum cut of G; otherwise, return any singleton cut v with $\deg(v) = d_{\mathsf{min}}$.

The space complexity of Algorithm 1 can be bounded as follows.

Claim 2.1. *Algorithm 1 always uses space of $O(n \log n)$ bits.*

Proof. The first pass of the algorithm requires storing two edges per vertex—each edge can be sampled using reservoir sampling using $O(\log n)$ bits. We also need to maintain a counter per vertex to compute d_{min} which again can be done in $O(\log n)$ bits per vertex.

The second pass involves computing each F_i as a spanning forest of $H \setminus F_1 \cup \cdots \cup F_{i-1}$ for all $i \in [d_{\mathsf{min}}]$. Since H always has $t = O(n/d_{\mathsf{min}})$ vertices (otherwise the algorithm aborts), each F_i is going to have $O(n/d_{\mathsf{min}})$ edges and thus we also need to store $O(n)$ edges in total in the second pass which requires $O(n \log n)$ bits. This implies that the overall space needed by the algorithm is $O(n \log n)$ bits. ∎

We now analyze the correctness of the algorithm. Each of the two subsequent lemmas identify the main property of each pass of the algorithm. In what follows, let $C := \{e_1, \ldots, e_\lambda\}$ denote any arbitrary minimum *non-singleton* cut of G (we thus slightly abuse the notation and use λ to denote the value of minimum non-singleton cuts and *not* necessarily the minimum cut of G).

Lemma 2.2. *With probability $\Omega(1)$, in the graph $G^{(2)}$ at the end of the first pass: (i) number of connected components is at most $100 \cdot n/d_{\mathsf{min}}$; and (ii) if $\lambda < d_{\mathsf{min}}$, then no edge $e_1, \ldots, e_\lambda$ in C has both endpoints in the same connected component.*

Lemma 2.2 already follows from the results of [15] (see Theorem 2.4 in the arXiv version). To keep this note self-contained, we give a complete and different proof of this lemma.

Lemma 2.3. *Conditioned on the event in Lemma 2.2, the cut output by the algorithm in the second pass is a minimum cut of G.*

We prove these lemmas in the next two subsections and for now show how they imply Theorem 1.

Proof of Theorem 1. By Lemmas 2.2 and 2.3, Algorithm 1 already outputs a minimum cut with probability $\Omega(1)$ in two passes and $O(n \log n)$ space. The remaining observation is that this algorithm also returns the value of whatever cut it finds correctly as the cuts in H correspond to cuts in G. As such, we can run Algorithm 1 in parallel $O(1)$ times and return the cut with the smallest number of edges as the final answer, to boost the success probability to any desired constant. ∎

First Pass: Proof of Lemma 2.2

Part (i): Bound on the number of connected components.

Proof Strategy: We first give a high-level overview of the proof strategy. The goal will be to upper bound the probability that $G^{(2)}$ has more than $100 \cdot n/d_{\mathsf{min}}$ connected components. For this event to happen, $G^{(2)}$ necessarily needs to have at least $k := 50 \cdot n/d_{\mathsf{min}}$ connected components, each of size at most $d_{\mathsf{min}}/50$; otherwise the remaining components will have more than n vertices together, a contradiction. So the goal now becomes bounding the probability that $G^{(2)}$ has at least k connected components of size at most $d_{\mathsf{min}}/50$. The proof is now by a careful union bound argument, over all possible "counter examples", namely, all ways of creating k such connected components for $G^{(2)}$. To do this, we first observe that each such component can be associated with a BFS tree of size at most the size of the component. Next, we observe that if a fixed tree of size s is indeed the associated BFS tree, then at least $s+1$ edges of the component are between vertices of the BFS tree. However, this event is unlikely since the sizes of the components are small (at most $d_{\mathsf{min}}/50$), and each vertex samples edges uniformly from all its neighbors (that are at least d_{min} in number.) Finally, we give a crude upper bound on the number of BFS trees of size s, which, together with a union bound, tells us that for a fixed s the probability of getting a BFS tree of size s is small. From the discussion above, this is also an upper bound on the probability that we get a component of size at most s, which will allow us to finalize the proof.

Details of the proof. As mentioned in the proof sketch, we want to upper bound the probability that $G^{(2)}$ has at least $k := 50 \cdot n/d_{\mathsf{min}}$ connected components of size at most $d_{\mathsf{min}}/50$. Consider the directed graph D obtained by directing each edge of $G^{(2)}$ from the vertex that sampled it to the other endpoint. The above bound implies that there are at least k vertices in D such that if we perform a BFS from them, we can only reach $d_{\mathsf{min}}/50$ vertices and these BFS trees are disjoint across the k vertices (this is true for undirected edges and thus certainly true after we direct the edges). We now bound the probability that this event happens.

Let $v_1, \ldots, v_k$ be any arbitrary set of k vertices from D. Let s be any integer between k and n and consider any choice of $s_1 + \ldots + s_k = s$ where $1 \leq s_i \leq d_{\mathsf{min}}/50$. Define the following event:

- $\mathcal{E}(v_1, \ldots, v_k, s_1, \ldots, s_k)$: The BFS tree starting from each v_i in the graph D has s_i vertices and is disjoint from the BFS trees of all v_j for $j \neq i$.

By the above discussion and a union bound,

$$\Pr\left(G^{(2)} \text{ has } > 100 \cdot n/d_{\mathsf{min}} \text{ components}\right) \leq \sum_{\substack{\text{all valid choices of} \\ v_1, \ldots, v_k \text{ and } s_1, \ldots, s_k}} \Pr\left(\mathcal{E}(v_1, \ldots, v_k, s_1, \ldots, s_k)\right). \tag{1}$$

We bound this probability in the following, starting with the following key claim.

Claim 2.4. *For any valid choices of $v_1, \ldots, v_k$ and $s_1, \ldots, s_k$ with $s_1 + \cdots + s_k = s$,*

$$\Pr\left(\mathcal{E}(v_1, \ldots, v_k, s_1, \ldots, s_k)\right) \leq \left(\frac{3}{50}\right)^s \cdot \left(\frac{50}{d_{\mathsf{min}}}\right)^k.$$

Proof. Let us fix some v_i and s_i for $i \in [k]$ and bound the probability that the BFS tree of v_i has size s_i. We first pick a "skeleton" of the BFS tree: consider any sequence $(x_1, \ldots, x_{s_i})$ of integers where $0 \leq x_j \leq 2$ and $x_1 + \ldots + x_{s_i} = s_i - 1$. We interpret this sequence as the number of child-nodes of vertices in the actual BFS tree starting from v_i where the nodes are written in the order that they are dequeued in the BFS traversal (see Figure 1 for an illustration). Note that this skeleton does *not* fix which vertices of D belong to the tree (beside v_i) but only the number of outgoing edges each of them may have in the tree.

Let us also fix a skeleton $(x_1, \ldots, x_{s_i})$. In this tree, having a node with in-degree < 2 necessarily means that the corresponding sampled edge of this node ends in an already visited node of the tree (and hence the new edge is not part of the BFS tree). Considering there are $s_i - 1$ edges in this tree but each of the s_i nodes sampled 2 edges in the algorithm, we have that $s_i + 1$ sampled edges ends up in an already visited node. Regardless of the choice of which actual vertices in D belong to the tree, the probability that this event happens for an edge is at most (s_i/d_{min}) (note that the actual probability depends on the vertices included in the skeleton but it is always $\leq s_i/d_{\mathsf{min}}$). As such,

$$\Pr\left(\text{BFS tree starting at } v_i \text{ in } D \text{ has the skeleton } (x_1, \ldots, x_{s_i})\right) \leq \left(\frac{s_i}{d_{\mathsf{min}}}\right)^{s_i+1}.$$

As the total number of possible skeletons with s_i vertices is at most 3^{s_i}, we have,

$$\Pr\left(\text{BFS tree starting at } v_i \text{ has } s_i \text{ vertices }\right) \leq 3^{s_i} \cdot \left(\frac{s_i}{d_{\mathsf{min}}}\right)^{s_i+1} \leq \left(\frac{3 \cdot s_i}{d_{\mathsf{min}}}\right)^{s_i},$$

as $0 < s_i < d_{\mathsf{min}}$. As the choice of vertices across BFS trees for $v_1, \ldots, v_k$ are disjoint, the events above independently happen for each one and thus,

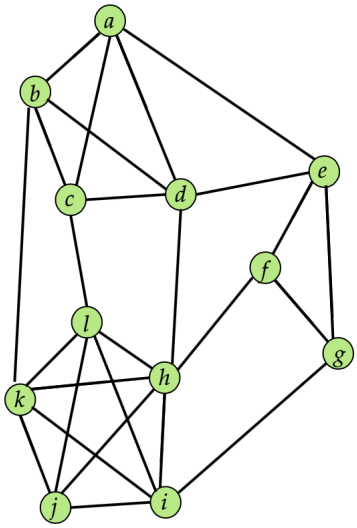
$$\begin{aligned}
\Pr\left(\mathcal{E}(v_1, \ldots, v_k, s_1, \ldots, s_k)\right) &\leq \prod_{i=1}^{k} \left(\frac{3 \cdot s_i}{d_{\mathsf{min}}}\right)^{s_i} = \left(\frac{3}{d_{\mathsf{min}}}\right)^{s} \cdot \prod_{i=1}^{k} s_i^{s_i} \\
&\leq \left(\frac{3}{d_{\mathsf{min}}}\right)^{s} \cdot \left(\frac{d_{\mathsf{min}}}{50}\right)^{s-k} \leq \left(\frac{3}{50}\right)^{s} \cdot \left(\frac{50}{d_{\mathsf{min}}}\right)^{k}, \\
&\qquad \text{(as } 1 \leq s_i \leq d_{\mathsf{min}}/50 \text{ and so at most } s-k \text{ indices can have value other than 1)}
\end{aligned}$$

concluding the proof. ∎

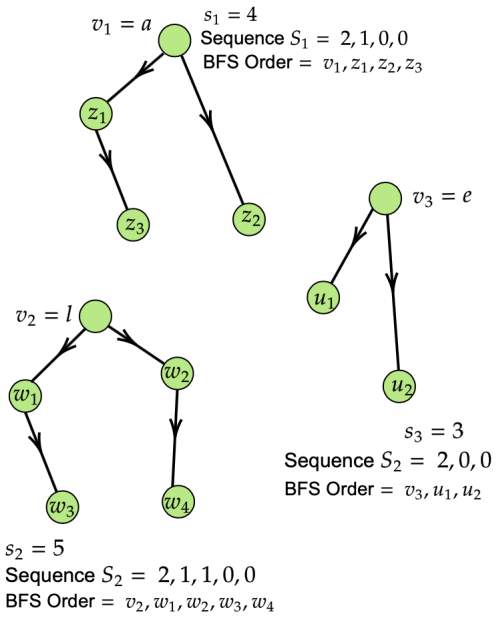
We can now conclude the proof by simply counting all possible choices of $v_1, \ldots, v_k$ and $s_1, \ldots, s_k$ in Eq (1) and apply Claim 2.4 to each choice, as follows:

$$\begin{aligned}
\Pr\left(G^{(2)} \text{ has } > 100 \cdot n/d_{\mathsf{min}} \text{ components}\right) &\leq \binom{n}{k} \cdot \sum_{s=k}^{n} \binom{s+k}{k} \cdot \left(\frac{3}{50}\right)^{s} \cdot \left(\frac{50}{d_{\mathsf{min}}}\right)^{k} \\
&\qquad \text{(since there are at most } \tbinom{s+k}{k} \text{ choices for } s_1 + \ldots + s_k = s) \\
&\leq \binom{n}{k} \cdot \left(\frac{k}{n}\right)^{k} \cdot \sum_{s=k}^{n} \left(\frac{12}{50}\right)^{s} \\
&\qquad \text{(as } 50/d_{\mathsf{min}} = k/n \text{ and } \tbinom{s+k}{k} \leq 2^{s+k} \leq 2^{2s} = 4^s) \\
&\leq \sum_{s=k}^{n} \left(\frac{36}{50}\right)^{s} \qquad \text{(as } \tbinom{n}{k} \leq e^k \cdot (n/k)^k \text{ and } e^k \leq 3^s) \\
&< 10^{-5},
\end{aligned}$$

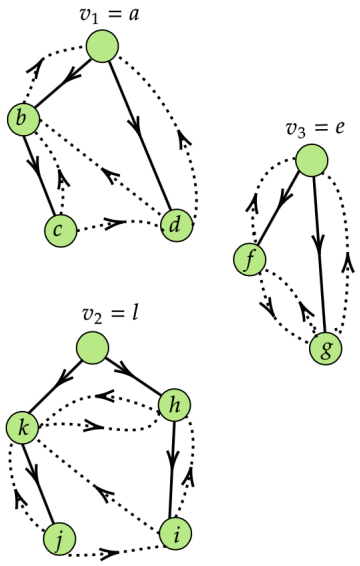
as $k \geq 50$. As such, with probability at least $1 - 10^{-5}$, number of components is at most $100 \cdot n/d_{\mathsf{min}}$.



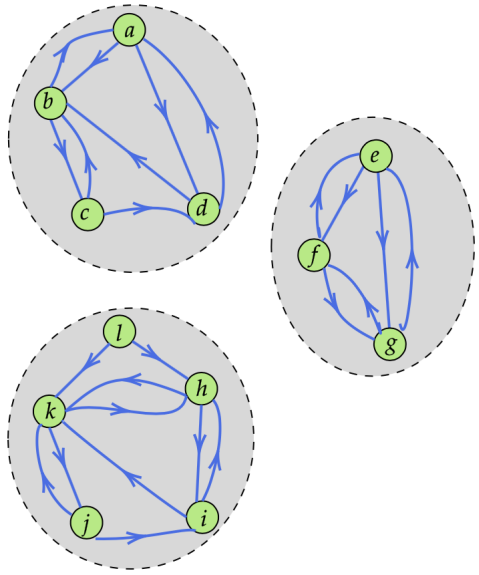
(a) The original graph G.



(b) Skeletons of sequences S_1, S_2, and S_3.



(c) Choosing vertices that belong in the BFS trees of v_1, v_2 and v_3. The dotted edges denote other edges sampled by these vertices.



(d) One way of sampling edges so that $G^{(2)}$ has components of size $4, 5$ and 3.

Figure 1: An illustration of the proof of Claim 2.4. In this example, we consider the case when $k = 3$, $s_1 = 4, s_2 = 5, s_3 = 3$, and $v_1 = a, v_2 = l, v_3 = e$. In **(a)**, we have the original graph G, from which $G^{(2)}$ is being sampled. In **(b)**, we choose one choice of sequences S_i that are of length s_i for $i \in [3]$, and the sum of the numbers in S_i is $s_i - 1$; the figure now show examples of "skeletons" consistent with sequences S_i. In **(c)** we choose one assignment of nodes to the skeletons in $G^{(2)}$, and we get BFS trees consistent with the sequence. Observe that the same skeleton could give rise to a different BFS tree depending on the assignment of the nodes. The dotted lines show one choice of remaining edges that have been sampled. These edges necessarily point to a vertex already in the tree. Finally, in **(d)**, we show the connected components of $G^{(2)}$ that are consistent with these choices.

Part (ii): Preserving the cut $C = \{e_1, \ldots, e_\lambda\}$ when $\lambda < d_{\mathsf{min}}$. Let $N(C)$ denote the vertices incident on $e_1, \ldots, e_\lambda$ and for each $v \in N(C)$, $c(v)$ denote the number of edges incident on v that belong to the cut C. An important observation is that for every vertex $v \in N(C)$, $c(v) \leq \deg(v)/2$; otherwise, by moving v to the other side of the cut (which is a valid move as this is a non-singleton cut), we obtain a cut with strictly smaller number of edges, a contradiction.

The probability that no edge $e_1, \ldots, e_\lambda$ has both its endpoints in the same component of $G^{(2)}$ is equal to the probability that none of these edges are sampled in $G^{(2)}$ which is equal to:

$$\prod_{v \in N(C)} \left(1 - \frac{c(v)}{\deg(v)}\right)^2 \geq \exp\left(-4 \sum_{v \in N(C)} \frac{c(v)}{\deg(v)}\right)$$
$$\text{(as } c(v)/\deg(v) \leq 1/2 \text{ and } 1 - x \geq e^{-2x} \text{ for } x \in [0, 1/2]\text{)}$$
$$\geq \exp\left(-\frac{4}{\lambda} \sum_{v \in N(C)} c(v)\right) = \exp\left(-\frac{8\lambda}{\lambda}\right) = e^{-8}.$$

To conclude, the probability that both events in part (i) and part (ii) happen simultaneously is at least $e^{-8} - 10^{-5} > e^{-9}$, finalizing the proof of Lemma 2.2. ∎

Second Pass: Proof of Lemma 2.3

Firstly, since H is obtained from G by contraction, any cut in H corresponds to some cut in G and thus minimum cut of H is at least as large as that of G. Suppose H is k-edge-connected. Then, any cut in H has at least k edges (by definition) and subsequently any cut in F has at least $\min\{d_{\mathsf{min}}, k\}$ edges by the choice of spanning forests $F_1, \ldots, F_{d_{\mathsf{min}}}$: consider any cut with less than $\min\{d_{\mathsf{min}}, k\}$ edges in F and let e be an edge of this cut in H but not in F; the reason e is not included in F by the algorithm can only be that e creates a cycle in every one of edge-disjoint spanning forests $F_1, \ldots, F_{d_{\mathsf{min}}}$ in F but this can only happens if F already contains at least d_{min} edges of the cut (one per each spanning forest), a contradiction.

If $\lambda < d_{\mathsf{min}}$ and conditioned on the event of Lemma 2.2, H will be λ-connected and thus F is also λ-connected. The minimum cut in F then corresponds to a minimum cut in H which subsequently is a minimum cut in G, concluding the proof in this case.

If $\lambda \geq d_{\mathsf{min}}$, then H will be at least d_{min}-connected and thus F is also d_{min}-connected and we simply return a singleton cut of size d_{min} in G which is again the correct answer.

This concludes the proof of the lemma and thus Theorem 1. ∎

Remark 2.5. Our algorithm in Theorem 1 can also be implemented in two passes over a *dynamic* stream (with edge insertions and deletions) using $O(n \cdot \text{polylog}(n))$ space: In the first pass, we use standard ℓ_0-samplers [21] to sample the edges of each vertex, and in the second pass, we run the algorithm of [6] for finding a k-edge connected subgraph of an n-vertex graph in $\widetilde{O}(nk)$ space (for us, $k = d_{\mathsf{min}}$ and number of vertices is $O(n/d_{\mathsf{min}})$ which translates to an $\widetilde{O}(n)$ space as desired).

3 A Lower Bound for Global Minimum Cut

In this section, we prove Theorem 2. That is, we prove that any streaming algorithm that computes exactly, the minimum cut of an n-vertex graph must use space $\Omega(n \log\log n)$.

The general principle behind the lower bound is simply as follows: we create a bipartite graph in which the minimum cut is a simpleton cut corresponding to the minimum degree vertex on one

side of the bipartition—the goal of the streaming algorithm would then be to simply compute the minimum degree vertex on one side of the bipartite graph. We show that this requires the algorithm to maintain a non-trivial information for each vertex of the bipartition and use this to establish the lower bound. The formal proof is by a reduction from the following communication problem.

Definition 3.1. *In the* M-Fold-GreaterThan *problem for any integers* $M, N \geq 1$*, Alice and Bob are each given* M *separate* N*-bit numbers* $X = (x^1, \ldots, x^M)$ *and* $Y = (y^1, \ldots, y^M)$ *(each* $x^i, y^i \in \{0,1\}^N$*) and the goal is to determine the value of* $\mathrm{GT}_N^M(X, Y)$ *defined as follows:*

$$\mathrm{GT}_N^M(X,Y) = \begin{cases} 1 & \text{if } \exists\ x^i, y^i \text{ such that } x^i > y^i \\ 0 & \text{otherwise} \end{cases}.$$

Proposition 3.2. *For any integers* $M, N, r \geq 1$*, any* r*-round communication protocol for the* M-Fold-GreaterThan *problem, wherein Alice and Bob only send* r *messages to each other, requires* $\Omega_r(M \cdot N^{1/r})$ *bits of communication to succeed with constant probability more than half.*

We will first a reduction to the problem, and then prove Proposition 3.2. Essentially we show that any streaming algorithm for the min-cut problem that takes space c gives us a communication protocol for M-Fold-GreaterThan that has cost c. Thus, in order to show a space lower bound for a streaming algorithm for the global minimum cut problem, it is sufficient to show a communication lower bound for M-Fold-GreaterThan.

Lemma 3.3. *Let* M *and* N *be positive integers such that* $M = 8 \cdot 2^N$*. Let* $\mathcal{A}$ *be a* δ*-error* p*-pass streaming algorithm that determines if the global min-cut of a* $(2M)$*-vertex graph is* $\geq 2^N$ *or* $< 2^N$*. Then, there is a* δ*-error* $(2p-1)$*-round protocol* π *that solves* GT_N^M *with communication cost* $O(p \cdot s)$*, where* s *is the space needed by* $\mathcal{A}$*.*

Proof of Lemma 3.3. Given an instance (X, Y) of GT_N^M, Alice and Bob create a graph G with vertices $V = \{u_1, \cdots, u_M, v_1, \cdots, v_M\}$, and edges E_A, E_B (as functions of X and Y, respectively, and known only to the respective player), plus some extra input-independent edges (independent of X,Y and known by both parties). These edges are as follows (see Figure 2 for an illustration):

(*i*) $\boldsymbol{E_A}$**:** for $i \in [M]$, Alice adds $2^N - x^i$ edges from u_i to $2^N - x^i$ arbitrary vertices in $\{v_1, \cdots, v_M\}$;

(*ii*) $\boldsymbol{E_B}$**:** for $i \in [M]$, Bob adds y^i edges from u_i to y^i arbitrary vertices in $\{v_1, \cdots, v_M\}$;

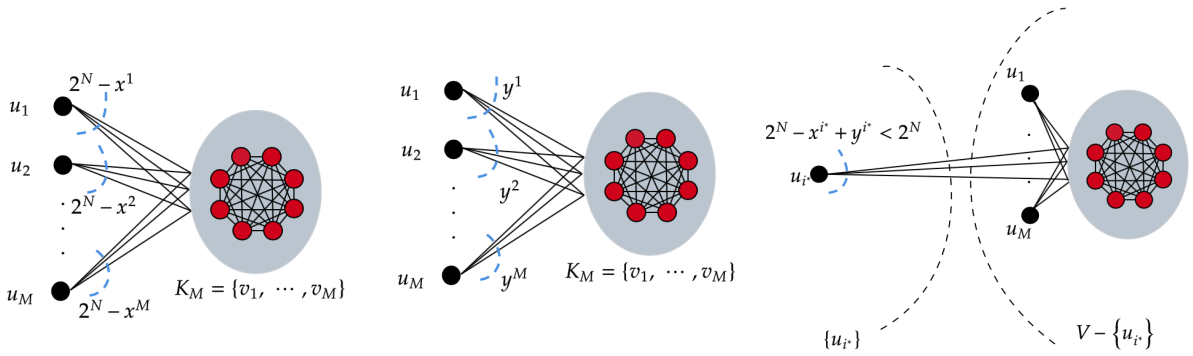
(*iii*) **Input-Independent edges:** Alice and Bob create a clique on $v_1, v_2, \cdots, v_M$.

The following claim states the main property of this graph.

Claim 3.4. *The minimum cut in the graph* G *has value equal to* $\min_{i \in [M]} \{y^i + 2^N - x^i\}$.

Proof. To prove this claim, we analyze the values of different cuts in the graph:

1. Any cut S where $\{v_1, \cdots, v_M\} \subseteq S$. In this case, the value of the cut is $\sum_{u_i \in V \setminus S} (2^N - x^i + y^i)$, which is minimized when $V \setminus S = \{u_{i^*}\}$ for $i^* = \mathrm{argmin}_{i \in [M]} \{2^N - x^i + y^i\}$.
2. Any cut S which partitions $\{v_1, \cdots, v_M\}$. Without loss of generality, let us assume S contains at least half (but not all) the vertices in $\{v_1, \cdots, v_M\}$ (otherwise, we can simply consider the $V \setminus S$ instead). Consider any $v_j \in V \setminus S$. Then this vertex has degree at least $M/2$ in S. Therefore, there are at least $M/2 \geq 4 \cdot 2^N$ edges going across such cuts.



(a) Edges E_A added by Alice. **(b)** Edges E_B added by Bob. **(c)** A min-cut of $G = (V, E_A \cup E_B)$.

Figure 2: An illustration of the graphs created by Alice and Bob in Lemma 3.3. Alice and Bob add edges E_A and E_B that depend on $X = \{x^1, \cdots, x^M\}$ and $Y = \{y^1, \cdots, y^M\}$, respectively. The min-cut of $G = (V, E_A \cup E_B)$ is $(\{u_{i^*}\}, V \setminus \{u_{i^*}\})$, where $i^* = \text{argmin}_{i \in [M]} \{2^N - x^i + y^i\}$. It has value less than $2^N \Leftrightarrow \text{GT}_N^M(X, Y) = 1$.

Note that for any $i \in [M]$, $2^N - x^i + y^i \leq 2^{N+1} - 1$ considering $0 \leq y^i, x^i \leq 2^N - 1$. Consequently, cuts of type $(\{u_{i^*}\}, V \setminus \{u_{i^*}\})$ in case 1 have a smaller value than those described in case 2. This implies that the minimum cut in the graph indeed has value $\min_{i \in [M]} \{2^N - x^i + y^i\}$. ▮ Claim 3.4

By Claim 3.4, the value of the minimum cut in G determines $\text{GT}_N^M(X, Y)$ as well:

(*i*) Suppose that the minimum cut of G is at least 2^N. Therefore, for all $i \in [M]$, $y^i + 2^N - x^i \geq 2^N$. This implies that for all $i \in [M]$, $x^i \leq y^i$, that is $\text{GT}_N^M(X, Y) = 0$.

(*ii*) On the other hand, suppose the minimum cut of G is less than 2^N. By Claim 3.4, there is some $i^* \in [M]$ with $2^N - x^{i^*} + y^{i^*} < 2^N$, implying that $x^{i^*} > y^{i^*}$; thus, $\text{GT}_N^M(X, Y) = 1$.

The final step is a standard simulation of the streaming $\mathcal{A}$ on the graph G in a communication and round efficient manner. Alice now runs $\mathcal{A}$ on her part of the graph and sends the contents of the memory to Bob, and Bob continues running $\mathcal{A}$ on his part of the graph (either player can run $\mathcal{A}$ on input-independent edges). This corresponds to the first pass of $\mathcal{A}$ on the stream $E_A \cup E_B$. Bob then sends the contents of the memory to Alice, who continues the execution of $\mathcal{A}$ (second pass). In the last pass, Bob outputs 0 if $\mathcal{A}$ returns that the minimum cut is at least 2^N and 1 otherwise.

Since Alice and Bob simply simulate $\mathcal{A}$ and send the memory contents as messages to each other, the protocol also errs with probability at most δ. Further, each pass of $\mathcal{A}$ is simulated by first Alice sending a message to Bob, and Bob sending a message back to Alice. In the final pass, Bob simply returns the answer. Therefore, there are at most $2p - 1$ messages sent in the protocol, and its communication cost if $O(s \cdot p)$, where s is the space used by $\mathcal{A}$, as in each round, the players are only communicating contents of the memory of $\mathcal{A}$ which is of size s at most. This completes the proof of Lemma 3.3. ▮

We now move on to the proof of Proposition 3.2. Towards this, we introduce some notation.

Definition 3.5. *For a communication problem P, we define $\text{R}_\delta(\pi)$ to be the minimum communication complexity of a δ-error randomized protocol π that solves P.*

Definition 3.6. *Let P be a communication problem. Let $\mathcal{D}$ be a distribution from which the instances of problem P are drawn. Then, $\text{D}_{\mathcal{D}}^{\delta}(\pi)$ is the minimum communication cost of a δ-error protocol π over the distribution $\mathcal{D}$.*

We denote the *Shannon entropy* of a random variable A by $\mathbb{H}(\mathsf{A})$. The mutual information of two random variables A and B is denoted by $\mathbb{I}(\mathsf{A} : \mathsf{B}) = \mathbb{H}(\mathsf{A}) - \mathbb{H}(\mathsf{A} \mid \mathsf{B})$.

Definition 3.7. *Consider a distribution $\mathcal{D}$ and a protocol π for some problem P. Let $(\mathsf{A}, \mathsf{B}) \sim \mathcal{D}$ be the input. Let $\Pi = \Pi(\mathsf{A}, \mathsf{B})$ denote the transcript of the protocol concatenated with public randomness used by π. Then, we define $\mathrm{ICost}_{\mathcal{D}}(\pi)$ of a protocol π with respect to $\mathcal{D}$ to be $\mathbb{I}_{\mathcal{D}}(\Pi : \mathsf{A} \mid \mathsf{B}) + \mathbb{I}_{\mathcal{D}}(\Pi : \mathsf{A} \mid \mathsf{B})$.*

Definition 3.8. *We define $\mathrm{IC}^{\delta}_{\mathcal{D}}(P)$ of a problem P with respect to a distribution $\mathcal{D}$ is the minimum $\mathrm{ICost}_{\mathcal{D}}(\pi)$ taken over all δ-error protocols π.*

We use the following basic properties of entropy and mutual information. We refer the reader to Chapter 2 of [12].

Fact 3.9. *Let A, B and C be three random variables that may or may not be correlated:*

1. *$0 \leq \mathbb{H}(\mathsf{A}) \leq \log |\mathsf{A}|$, where $|\mathsf{A}|$ be the size of support of A. The equality holds iff A is uniformly distributed on its support.*
2. *$\mathbb{I}(\mathsf{A} : \mathsf{B}) \geq 0$. Equality holds iff $\mathsf{A} \perp \mathsf{B}$.*
3. *Conditioning on a random variable reduces entropy: $\mathbb{H}(\mathsf{A} \mid \mathsf{B}, \mathsf{C}) \leq \mathbb{H}(\mathsf{A} \mid \mathsf{B})$. Equality holds iff $\mathsf{A} \perp \mathsf{C} \mid \mathsf{B}$.*
4. *Chain rule for mutual information: $\mathbb{I}(\mathsf{A}, \mathsf{B} : \mathsf{C}) = \mathbb{I}(\mathsf{A} : \mathsf{C}) + \mathbb{I}(\mathsf{B} : \mathsf{C} \mid \mathsf{A})$.*

Fact 3.10. *For any random variables A, B and C, $\mathbb{I}(\mathsf{A} : \mathsf{B} \mid \mathsf{C}) \leq \mathbb{I}(\mathsf{A} : \mathsf{B}) + \mathbb{H}(\mathsf{C})$.*

Proof.

$$\begin{aligned}
\mathbb{I}(\mathsf{A} : \mathsf{B} \mid \mathsf{C}) &= \mathbb{I}(\mathsf{A} : \mathsf{B}, \mathsf{C}) - \mathbb{I}(\mathsf{A} : \mathsf{C}) \\
&\quad \text{(follows from Fact 3.9-(4))} \\
&= \mathbb{I}(\mathsf{A} : \mathsf{B}) + \mathbb{I}(\mathsf{A} : \mathsf{C} \mid \mathsf{B}) - \mathbb{I}(\mathsf{A} : \mathsf{C}) \\
&\quad \text{(follows from Fact 3.9-(4))} \\
&\leq \mathbb{I}(\mathsf{A} : \mathsf{B}) + \mathbb{H}(\mathsf{C} \mid \mathsf{B}) \\
&\quad \text{(follows from definition of mutual information and Fact 3.9-(2))} \\
&\leq \mathbb{I}(\mathsf{A} : \mathsf{B}) + \mathbb{H}(\mathsf{C}) \\
&\quad \text{(Conditioning only decreases entropy).}
\end{aligned}$$

∎

Fact 3.11. *For random variables $\mathsf{A}, \mathsf{B}, \mathsf{C}$ and D if $\mathsf{A} \perp \mathsf{D} \mid \mathsf{C}$, then $\mathbb{I}(\mathsf{A} : \mathsf{B} \mid \mathsf{C}) \leq \mathbb{I}(\mathsf{A} : \mathsf{B} \mid \mathsf{C}, \mathsf{D})$.*

Proof. Since A and D are independent conditioned on C, by Fact 3.9-(2), $\mathbb{H}(\mathsf{A} \mid \mathsf{C}) = \mathbb{H}(\mathsf{A} \mid \mathsf{C}, \mathsf{D})$ and $\mathbb{H}(\mathsf{A} \mid \mathsf{C}, \mathsf{B}) = \mathbb{H}(\mathsf{A} \mid \mathsf{C}, \mathsf{B}, \mathsf{D})$. Consequently,

$$\begin{aligned}
\mathbb{I}(\mathsf{A} : \mathsf{B} \mid \mathsf{C}) &= \mathbb{H}(\mathsf{A} \mid \mathsf{C}) - \mathbb{H}(\mathsf{A} \mid \mathsf{C}, \mathsf{B}) \\
&= \mathbb{H}(\mathsf{A} \mid \mathsf{C}, \mathsf{D}) - \mathbb{H}(\mathsf{A} \mid \mathsf{C}, \mathsf{B}) \\
&\leq \mathbb{H}(\mathsf{A} \mid \mathsf{C}, \mathsf{D}) - \mathbb{H}(\mathsf{A} \mid \mathsf{C}, \mathsf{B}, \mathsf{D}) \\
&= \mathbb{I}(\mathsf{A} : \mathsf{B} \mid \mathsf{C}, \mathsf{D})
\end{aligned}$$

∎

To prove this Proposition 3.2, we will give a hard distribution for GT_N^M and use Yao's lemma to prove a lower bound on $\mathrm{R}_\delta(\mathrm{GT}_N^M)$.

We state a known hard distribution for $\mathrm{GT}_N^1 = \mathrm{GT}_N$ that we will be using :

Distribution $\mathcal{D}_N$: A hard distribution for GT_N.

1. Sample an index $k \in \{1, 2, \cdots, N\}$ uniformly at random.
2. Sample $z_1, \cdots, z_{k-1}, w, x_{k+1}, \cdots, x_N, y_{k+1}, \cdots, y_N$ uniformly at random from $\{0, 1\}$.
3. Let $x = z_1, \cdots, z_{k-1}, w, x_{k+1} \cdots, x_N$ and let $y = z_1, \cdots, z_{k-1}, \bar{w}, y_{k+1}, \cdots, y_N$.

We define $\mathcal{D}_N^0 = \mathcal{D}_N \mid w = 0$ and $\mathcal{D}_N^1 = \mathcal{D}_N \mid w = 1$.

We state a result due to [11] about distribution $\mathcal{D}_N$.

Lemma 3.12. *Consider any deterministic ϵ-error protocol π for GT_N, where $\epsilon < \frac{1}{2}$, then, $\mathrm{ICost}_{\mathcal{D}_N}(\pi) = \Omega(\log N)$.*

Next we state a lemma due to [17] that allows us to give a bound on $\mathrm{ICost}_{\mathcal{D}_N^1}(\pi)$.

Lemma 3.13. *Fix any communication problem P. Let ϵ_1 and ϵ_2 be constants such that $0 < \epsilon_1 < \epsilon_2 < \frac{1}{2}$. Let $\mathcal{D}$ be the input distribution. For every ϵ_1 error protocol π for P on $\mathcal{D}$, there is an ϵ_2-error protocol π' for P on distribution $\mathcal{D}$ such that:*

$$\mathrm{ICost}_{\mathcal{D}}(\pi') = O(\mathrm{ICost}_{\mathcal{D}^1}(\pi) + \log \|\pi\|)$$

Consequently, we have the following corollary:

Corollary 3.14. *For $\delta < \frac{1}{2}$, any δ-error protocol π for $\mathcal{D}_N$ with $\|\pi\| = o(N)$ has $\mathrm{ICost}_{\mathcal{D}_N^1}(\pi) = \Omega(\log N)$.*

We now describe our hard distribution for GT_N^M:

Distribution $\mu_{N\times M}$: A hard distribution for GT_N^M:

1. Pick $i^* \in [M]$ uniformly at random. Choose $(x, y) \sim \mathcal{D}_N$.
2. Let $(x_i, y_i) \sim \mathcal{D}_N^1$ for all $i \neq i^*$.
3. Let $(x_{i^*}, y_{i^*}) = (x, y)$.

We wish to prove the following theorem about $\mu_{N\times M}$:

Theorem 3. *Consider any δ-error protocol π for GT_N^M on the distribution $\mu_{N\times M}$. Then, there exists a protocol π' for GT_N on distribution $\mathcal{D}_N$ such that:*

1. $\frac{c}{M} \cdot \mathrm{ICost}_{\mu_{N\times M}}(\pi) \geq \mathrm{ICost}_{\mathcal{D}_N^1}(\pi')$, *where c is a constant, and*
2. $\|\pi\| = \|\pi'\|$

We first show that it is sufficient to prove an intermediate lemma:

Lemma 3.15. *Consider any δ-error protocol π for GT_N^M on the distribution $\mu_{N\times M}$. Then, there exists a protocol π' for GT_N on distribution $\mathcal{D}_N$ such that:*

1. $\frac{1}{M}\cdot \mathrm{ICost}_{\mu^1_{N\times M}}(\pi) \geq \mathrm{ICost}_{\mathcal{D}^1_N}(\pi')$, *and*
2. $\|\pi\| = \|\pi'\|$

Proof of Theorem 3. Consider any δ-error protocol π. Assuming that $\frac{1}{M}\cdot\mathrm{ICost}_{\mu^1_{N\times M}}(\pi) \geq \mathrm{ICost}_{\mathcal{D}^1_N}(\pi')$, we prove that $\frac{c}{M}\cdot \mathrm{ICost}_{\mu_{N\times M}}(\pi) \geq \mathrm{ICost}_{\mathcal{D}^1_N}(\pi')$, where c is a constant. We let $(\mathsf{A},\mathsf{B}) \sim \mu_{N\times M}$ and consider,

$$\begin{aligned}
\mathrm{ICost}_{\mu_{N\times M}}(\pi) &= \mathbb{I}_{\mu_{N\times M}}(\Pi : \mathsf{A} \mid \mathsf{B}) + \mathbb{I}_{\mu_{N\times M}}(\Pi : \mathsf{B} \mid \mathsf{A}) \\
&\geq \mathbb{I}_{\mu_{N\times M}}(\Pi : \mathsf{A} \mid \mathsf{B}, w) + \mathbb{I}_{\mu_{N\times M}}(\Pi : \mathsf{B} \mid \mathsf{A}, w) - 2\cdot\mathbb{H}(w) \\
&\quad \text{(Implied by Fact 3.10)} \\
&= \frac{1}{2}\cdot\mathbb{I}_{\mu_{N\times M}}(\Pi : A \mid B, w=1) + \frac{1}{2}\cdot\mathbb{I}_{\mu_{N\times M}}(\Pi : B \mid A, w=1) - 2\cdot\mathbb{H}(w) \\
&\quad \text{(by definition of mutual information)} \\
&= \frac{1}{2}\cdot\mathbb{I}_{\mu^1_{N\times M}}(\Pi : A \mid B) + \frac{1}{2}\cdot\mathbb{I}_{\mu^1_{N\times M}}(\Pi : B \mid A) - 2 \\
&\quad \text{(since } \mathbb{H}(w)=1\text{)} \\
&= \frac{1}{2}\cdot\mathrm{ICost}_{\mu^1_{N\times M}}(\pi) - 2.
\end{aligned}$$

∎

So, we are left with the task of proving Lemma 3.15. To do this, we state the following well-known claim that we will need to prove our claim:

Claim 3.16. *For any distribution $\mathcal{D}$ and any protocol π, let R be the public randomness used in π, then,* $\mathrm{ICost}_{\mathcal{D}}(\pi) = \mathbb{I}_{\mathcal{D}}(\Pi : \mathsf{A} \mid \mathsf{B}, R) + \mathbb{I}_{\mathcal{D}}(\Pi : \mathsf{B} \mid \mathsf{A}, R)$

Proof.

$$\begin{aligned}
\mathrm{ICost}_{\mathcal{D}}(\pi) &= \mathbb{I}(\Pi : \mathsf{A} \mid \mathsf{B}) + \mathbb{I}(\Pi : \mathsf{B} \mid \mathsf{A}) \\
&= \mathbb{I}(\Pi, R : \mathsf{A} \mid \mathsf{B}) + \mathbb{I}(\Pi, R : \mathsf{B} \mid \mathsf{A}) \\
&\quad (\Pi \text{ denotes the concatenation of the transcript and the public randomness}) \\
&= \mathbb{I}(\Pi : \mathsf{A} \mid \mathsf{B}, R) + \mathbb{I}(R : \mathsf{A} \mid \mathsf{B}) + \mathbb{I}(\Pi : \mathsf{B} \mid \mathsf{A}, R) + \mathbb{I}(R : \mathsf{B} \mid \mathsf{A}) \\
&\quad \text{(By chain rule, that is, Fact 3.9-(4))} \\
&\quad \mathbb{I}(\Pi : \mathsf{A} \mid \mathsf{B}, R) + \mathbb{I}(\Pi : \mathsf{B} \mid \mathsf{A}, R) \\
&\quad \text{(Follows from the fact that } R \perp \mathsf{A}, \mathsf{B}\text{, and Fact 3.9-(3)).}
\end{aligned}$$

This proves our claim. ∎

Proof of Lemma 3.15. Given a δ-error protocol π for GT_N^M on distribution $\mu_{N\times M}$, we give a protocol π' for GT_N:

Protocol π'. The protocol for solving GT_N using protocol π for GT_N^M.
Input: An instance $(X, Y) \sim \mathcal{D}_N$. **Output:** Yes if $X \geq Y$ and No otherwise.

1. Using public randomness, the players sample $i^* \in [M]$ uniformly at random. Let $X_{i^*} = X$ and $Y_{i^*} = Y$.
2. Using public randomness, sample $X_{<i^*}$ and $Y_{>i^*}$ from $\mathcal{D}_N^1$ independently.
3. Using private randomness, Alice samples $X_{>i^*}$ such that $(X_j, Y_j) \sim \mathcal{D}_N^1$ for all $j > i^*$ and Bob similarly samples $Y_{<i^*}$.
4. Let $X' = (X_1, \cdots, X_M)$ and $Y' = (Y_1, \cdots, Y_M)$. The players then output the result of protocol π on (X', Y').

We note that the distribution created by π' is exactly the distribution $\mu_{N\times M}$, and $\mathrm{GT}_N(X, Y) = 1$ iff $\mathrm{GT}_N^M(X', Y) = 1$. It follows that π' is a δ-error protocol.

In the equations that follow, let I be the random variable for i^*.

$$\begin{aligned}
\mathrm{ICost}_{\mathcal{D}_N^1}(\pi') &= \mathbb{I}_{\mathcal{D}_N^1}(\Pi' : X \mid Y, R) + \mathbb{I}_{\mathcal{D}_N^1}(\Pi' : Y \mid X, R) \\
&\quad \text{(From Claim 3.16)} \\
&= \mathbb{I}_{\mathcal{D}_N^1}(\Pi' : X \mid Y, R, \mathsf{I}) + \mathbb{I}_{\mathcal{D}_N^1}(\Pi' : Y \mid X, R, \mathsf{I}) \\
&\quad (\mathsf{I} \text{ is chosen using public randomness}) \\
&= \sum_{i=1}^{M} \Pr(\mathsf{I} = i)\left(\mathbb{I}_{\mathcal{D}_N^1}(\Pi' : X \mid Y, R, \mathsf{I} = i) + \mathbb{I}_{\mathcal{D}_N^1}(\Pi' : Y \mid X, R, \mathsf{I} = i)\right) \\
&\quad \text{(by definition of mutual information)} \\
&= \frac{1}{M}\sum_{i=1}^{M} \mathbb{I}_{\mathcal{D}_N^1}(\Pi' : X_i \mid Y_i, X_{<i}, Y_{>i}, I = i) + \mathbb{I}_{\mathcal{D}_N^1}(\Pi' : Y_i \mid X_i, X_{<i}, Y_{>i}, I = i) \\
&\quad (R = (X_{<i}, Y_{>i}, \mathsf{I})) \\
&= \frac{1}{M}\sum_{i=1}^{M} \mathbb{I}_{\mathcal{D}_N^1}(\Pi' : X_i \mid Y_i, X_{<i}, Y_{>i}) + \mathbb{I}_{\mathcal{D}_N^1}(\Pi' : Y_i \mid X_i, X_{<i}, Y_{>i})
\end{aligned}$$

The last equality follows from the fact that since $(X, Y) \sim \mathcal{D}_N^1$, all sets (X_j, Y_j) are chosen from $\mathcal{D}_N^1$ and are independent of the event $\mathsf{I} = i$. We continue our argument:

$$\begin{aligned}
\mathrm{ICost}_{\mathcal{D}_N^1}(\pi') &\leq \frac{1}{M}\sum_{i=1}^{M} \mathbb{I}_{\mathcal{D}_N^1}(\Pi' : X_i \mid Y', X_{<i}) + \mathbb{I}_{\mathcal{D}_N^1}(\Pi' : Y_i \mid X', Y_{>i}) \\
&\quad (X_i \perp Y^{<i} \mid Y' \text{ and } Y_i \perp X^{>i} \mid X', \text{ so we can apply Fact 3.11}) \\
&= \frac{1}{M}(\mathbb{I}_{\mathcal{D}_N^1}(\Pi' : X' \mid Y') + \mathbb{I}_{\mathcal{D}_N^1}(\Pi' : Y' \mid X')) \\
&\quad \text{(Chain rule of mutual information)} \\
&= \frac{1}{M}(\mathbb{I}_{\mu_{N\times M}^1}(\Pi : X' \mid Y') + \mathbb{I}_{\mu_{N\times M}^1}(\Pi : Y' \mid X')) \\
&= \frac{1}{M} \cdot \mathrm{ICost}_{\mu_{N\times M}^1}(\pi).
\end{aligned}$$

This proves our claim. ∎

Remark 3.17. Our lower bound in this section also implies that the **randomized communication complexity** of the minimum cut problem in the standard two-player model of Yao [34] (with no restriction on the number of communication rounds) is $\Omega(n \log \log n)$ bits. Previously, a lower bound of $\Omega(n \log n)$ bits was known for *deterministic* protocols [18] but for *randomized* protocols, the best lower bound was $\Omega(n)$ (a folklore lower bound based on set disjointness; see, e.g. [13, 33]).

The proof of Remark 3.17 is simply by noting that our approach lower bounds the two-player communication complexity of minimum cut on n-vertex graphs, by that of $\text{GT}_N^M(\cdot)$ for $M = \Theta(n)$ and $N = \Theta(\log n)$. Similar to Proposition 3.2 for bounded-round protocols, it is also known that for unbounded-round protocols, the communication complexity of this problem is $\Omega(M \cdot \log N)$ bits (this is obtained by using the $\Omega(\log N)$ information complexity lower bound of (1-fold) Greater-Than problem in [11] in the direct sum results of [8, 10] for the M-fold problem). Plugging these bounds in the reduction implies Remark 3.17.

References

[1] K. J. Ahn, G. Cormode, S. Guha, A. McGregor, and A. Wirth. Correlation clustering in data streams. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 2237–2246, 2015. 2

[2] K. J. Ahn and S. Guha. Graph sparsification in the semi-streaming model. In *Automata, Languages and Programming, 36th Internatilonal Colloquium, ICALP 2009, Rhodes, Greece, July 5-12, 2009, Proceedings, Part II*, pages 328–338, 2009. 1

[3] K. J. Ahn and S. Guha. Linear programming in the semi-streaming model with application to the maximum matching problem. In *Automata, Languages and Programming - 38th International Colloquium, ICALP 2011, Zurich, Switzerland, July 4-8, 2011, Proceedings, Part II*, pages 526–538, 2011. 2

[4] K. J. Ahn and S. Guha. Access to data and number of iterations: Dual primal algorithms for maximum matching under resource constraints. *ACM Trans. Parallel Comput.*, 4(4):17:1–17:40, 2018. 2

[5] K. J. Ahn, S. Guha, and A. McGregor. Analyzing graph structure via linear measurements. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 459–467, 2012. 2

[6] K. J. Ahn, S. Guha, and A. McGregor. Graph sketches: sparsification, spanners, and subgraphs. In *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2012, Scottsdale, AZ, USA, May 20-24, 2012*, pages 5–14, 2012. 7

[7] S. Assadi, Y. Chen, and S. Khanna. Polynomial pass lower bounds for graph streaming algorithms. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 265–276, 2019. 1

[8] Z. Bar-Yossef, T. S. Jayram, R. Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. In *43rd Symposium on Foundations of Computer Science (FOCS 2002), 16-19 November 2002, Vancouver, BC, Canada, Proceedings*, pages 209–218, 2002. 14

[9] R. Becker, A. Karrenbauer, S. Krinninger, and C. Lenzen. Near-optimal approximate shortest paths and transshipment in distributed and streaming models. In *31st International Symposium on Distributed Computing, DISC 2017, October 16-20, 2017, Vienna, Austria*, pages 7:1–7:16, 2017. 2

[10] M. Braverman. Interactive information complexity. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 505–524, 2012. 14

[11] M. Braverman and O. Weinstein. A discrepancy lower bound for information complexity. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques - 15th International Workshop, APPROX 2012, and 16th International Workshop, RANDOM 2012, Cambridge, MA, USA, August 15-17, 2012. Proceedings*, pages 459–470, 2012. 11, 14

[12] T. M. Cover and J. A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, USA, 2006. 10

[13] J. Feigenbaum, S. Kannan, A. McGregor, S. Suri, and J. Zhang. On graph problems in a semi-streaming model. *Theor. Comput. Sci.*, 348(2-3):207–216, 2005. 1, 14

[14] M. Ghaffari, T. Gouleakis, C. Konrad, S. Mitrovic, and R. Rubinfeld. Improved massively parallel computation algorithms for mis, matching, and vertex cover. In *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing, PODC 2018, July 23-27, 2018*, pages 129–138, 2018. 2

[15] M. Ghaffari, K. Nowicki, and M. Thorup. Faster algorithms for edge connectivity via random 2-out contractions. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 1260–1279, 2020. 1, 2, 3

[16] R. E. Gomory and T. C. Hu. Multi-terminal network flows. *Journal of the Society for Industrial and Applied Mathematics*, 9(4):551–570, 1961. 1

[17] M. Göös, T. S. Jayram, T. Pitassi, and T. Watson. Randomized communication vs. partition number. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, pages 52:1–52:15, 2017. 11

[18] A. Hajnal, W. Maass, and G. Turán. On the communication complexity of graph properties. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 186–191, 1988. 14

[19] J. Hao and J. B. Orlin. A faster algorithm for finding the minimum cut in a graph. In *Proceedings of the Third Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms, 27-29 January 1992, Orlando, Florida, USA*, pages 165–174, 1992. 1

[20] P. Indyk, S. Mahabadi, R. Rubinfeld, J. Ullman, A. Vakilian, and A. Yodpinyanee. Fractional set cover in the streaming model. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017, August 16-18, 2017, Berkeley, CA, USA*, pages 12:1–12:20, 2017. 2

[21] H. Jowhari, M. Saglam, and G. Tardos. Tight bounds for lp samplers, finding duplicates in streams, and related problems. In *Proceedings of the 30th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2011, June 12-16, 2011, Athens, Greece*, pages 49–58, 2011. 7

[22] M. Kapralov, Y. T. Lee, C. Musco, C. Musco, and A. Sidford. Single pass spectral sparsification in dynamic streams. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 561–570, 2014. 1

[23] D. R. Karger. Global min-cuts in rnc, and other ramifications of a simple min-cut algorithm. In *Proceedings of the Fourth Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms, 25-27 January 1993, Austin, Texas, USA*, pages 21–30, 1993. 1

[24] D. R. Karger. Random sampling in cut, flow, and network design problems. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, pages 648–657, 1994. 1

[25] D. R. Karger. Minimum cuts in near-linear time. *J. ACM*, 47(1):46–76, 2000. 1

[26] D. R. Karger and C. Stein. A new approach to the minimum cut problem. *J. ACM*, 43(4):601–640, 1996. 1

[27] K. Kawarabayashi and M. Thorup. Deterministic edge connectivity in near-linear time. *J. ACM*, 66(1):4:1–4:50, 2019. 1

[28] R. Kumar, B. Moseley, S. Vassilvitskii, and A. Vattani. Fast greedy algorithms in mapreduce and streaming. In *25th ACM Symposium on Parallelism in Algorithms and Architectures, SPAA '13, Montreal, QC, Canada - July 23 - 25, 2013*, pages 1–10, 2013. 2

[29] S. Lattanzi, B. Moseley, S. Suri, and S. Vassilvitskii. Filtering: a method for solving graph problems in mapreduce. In *SPAA 2011: Proceedings of the 23rd Annual ACM Symposium on Parallelism in Algorithms and Architectures, San Jose, CA, USA, June 4-6, 2011 (Co-located with FCRC 2011)*, pages 85–94, 2011. 2

[30] A. McGregor. Graph stream algorithms: a survey. *SIGMOD Rec.*, 43(1):9–20, 2014. 1

[31] S. Mukhopadhyay and D. Nanongkai. Weighted min-cut: sequential, cut-query, and streaming algorithms. In *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 496–509, 2020. 1

[32] A. Rubinstein, T. Schramm, and S. M. Weinberg. Computing exact minimum cuts without knowing the graph. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, pages 39:1–39:16, 2018. 1, 2

[33] X. Sun and D. P. Woodruff. Tight bounds for graph problems in insertion streams. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2015, August 24-26, 2015, Princeton, NJ, USA*, pages 435–448, 2015. 14

[34] A. C. Yao. Some complexity questions related to distributive computing (preliminary report). In *Proceedings of the 11h Annual ACM Symposium on Theory of Computing, April 30 - May 2, 1979, Atlanta, Georgia, USA*, pages 209–213, 1979. 14

[35] M. Zelke. Intractability of min- and max-cut in streaming graphs. *Inf. Process. Lett.*, 111(3):145–150, 2011. 1, 2