Distributed Minimum Cut Approximation

Mohsen Ghaffari[1] and Fabian Kuhn[2]

[1] Computer Science and Artificial Intelligence Lab, MIT, USA
ghaffari@mit.edu
[2] Department of Computer Science, University of Freiburg, Germany
kuhn@cs.uni-freiburg.de

Abstract. We study the problem of computing approximate minimum edge cuts by distributed algorithms. We use a standard synchronous message passing model where in each round, $O(\log n)$ bits can be transmitted over each edge (a.k.a. the CONGEST model). The first algorithm is based on a simple and new approach for analyzing random edge sampling, which we call the *random layering technique*. For any weighted graph and any $\epsilon \in (0,1)$, the algorithm with high probability finds a cut of size at most $O(\epsilon^{-1}\lambda)$ in $O(D) + \tilde{O}(n^{1/2+\epsilon})$ rounds, where λ is the size of the minimum cut and the $\tilde{O}$-notation hides poly-logarithmic factors in n. In addition, based on a centralized algorithm due to Matula [SODA '93], we present a randomized distributed algorithm that with high probability computes a cut of size at most $(2+\epsilon)\lambda$ in $\tilde{O}((D+\sqrt{n})/\epsilon^5)$ rounds for any $\epsilon > 0$.

The time complexities of our algorithms almost match the $\tilde{\Omega}(D+\sqrt{n})$ lower bound of Das Sarma et al. [STOC '11], thus leading to an answer to an open question raised by Elkin [SIGACT-News '04] and Das Sarma et al. [STOC '11].

To complement our upper bound results, we also strengthen the $\tilde{\Omega}(D+\sqrt{n})$ lower bound of Das Sarma et al. by extending it to unweighted graphs. We show that the same lower bound also holds for unweighted multigraphs (or equivalently for weighted graphs in which $O(w \log n)$ bits can be transmitted in each round over an edge of weight w). For unweighted simple graphs, we show that computing an α-approximate minimum cut requires time at least $\tilde{\Omega}(D+\sqrt{n}/\alpha^{1/4})$.

1 Introduction

Finding minimum cuts or approximately minimum cuts are classical and fundamental algorithmic graph problems with many important applications. In particular, minimum edge cuts and their size (i.e., the edge connectivity) are relevant in the context of networks, where edge weights might represent link capacities and therefore edge connectivity can be interpreted as the throughput capacity of the network. Decomposing a network using small cuts helps designing efficient communication strategies and finding communication bottlenecks (see, e.g., [20,27]). Both the exact and approximate variants of the minimum cut problem have received extensive attention in the domain of centralized algorithms (cf. Section 1.1 for a brief review of the results in the centralized setting). This line of research has led to (almost) optimal centralized algorithms with running times $\tilde{O}(m+n)$ [19] for the exact version and $O(m+n)$ [24] for constant-factor approximations, where n and m are the numbers of nodes and edges, respectively.

As indicated by Elkin [6] and Das Sarma et al. [4], the problem has remained essentially open in the distributed setting. In the LOCAL model [26] where in each round, a

Y. Afek (Ed.): DISC 2013, LNCS 8205, pp. 1–15, 2013.

message of unbounded size can be sent over each edge, the problem has a trivial time complexity of $\Theta(D)$ rounds, where D is the (unweighted) diameter of the network. The problem is therefore more interesting and also practically more relevant in models where messages are of some bounded size B. The standard model incorporating this restriction is the CONGEST model [26], a synchronous message passing model where in each time unit, B bits can be sent over every link (in each direction). It is often assumed that $B = \Theta(\log n)$. The only known non-trivial result is an elegant lower bound by Das Sarma et al. [4] showing that any α-approximation of the minimum cut in weighted graphs requires at least $\Omega(D + \sqrt{n/(B \log n)})$ rounds.

Our Contribution: We present two distributed minimum-cut approximation algorithms for undirected weighted graphs, with complexities almost matching the lower bound of [4]. We also extend the lower bound of [4] to unweighted graphs and multigraphs.

Our first algorithm, presented in Section 4, with high probability[1] finds a cut of size at most $O(\varepsilon^{-1}\lambda)$, for any $\epsilon \in (0,1)$ and where λ is the edge connectivity, i.e., the size of the minimum cut in the network. The time complexity of this algorithm is $O(D) + O(n^{1/2+\epsilon} \log^3 n \log\log n \log^* n)$. The algorithm is based on a simple and novel approach for analyzing random edge sampling, a tool that has proven extremely successful also for studying the minimum cut problem in the centralized setting (see, e.g., [20]). Our analysis is based on *random layering*, and we believe that the approach might also be useful for studying other connectivity-related questions. Assume that each edge $e \in E$ of an unweighted multigraph $G = (V, E)$ is independently sampled and added to a subset $E' \subset E$ with probability p. For $p \leq \frac{1}{\lambda}$, the graph $G' = (V, E')$ induced by the sampled edges is disconnected with at least a constant probability (just consider one min-cut). In Section 3, we use random layering to show that if $p = \Omega(\frac{\log n}{\lambda})$, the sampled graph G' is connected w.h.p. This bound is optimal and was known previously, with two elegant proofs: [23] and [15]. Our proof is simple and self-contained and it serves as a basis for our algorithm in Section 4.

The second algorithm, presented in Section 5, finds a cut with size at most $(2+\varepsilon)\lambda$, for any constant $\varepsilon > 0$, in time $O((D+\sqrt{n}\log^* n)\log^2 n \log\log n \cdot \frac{1}{\varepsilon^5})$. This algorithm combines the general approach of Matula's centralized $(2+\varepsilon)$-approximation algorithm [24] with Thurimella's algorithm for sparse edge-connectivity certificates [29] and with the famous random edge sparsification technique of Karger (see e.g., [16]).

To complement our upper bounds, we also extend the lower bound of Das Sarma et al. [4] to unweighted graphs and multigraphs. When the minimum cut problem (or more generally problems related to small edge cuts and edge connectivity) are in a distributed context, often the weights of the edges correspond to their capacities. It therefore seems reasonable to assume that over a link of twice the capacity, we can also transmit twice the amount of data in a single time unit. Consequently, it makes sense to assume that over an edge of weight (or capacity) $w \geq 1$, $O(w \log n)$ bits can be transmitted per round (or equivalently that such a link corresponds to w parallel links of unit capacity). The lower bound of [4] critically depends on having links with (very) large weight over which in each round only $O(\log n)$ bits can be transmitted. We generalize the approach of [4] and obtain the same lower bound result as in [4] for the weaker setting where edge weights correspond to edge capacities (i.e., the setting that can be modeled using

[1] We use the phrase *with high probability* (w.h.p.) to indicate probability greater than $1 - \frac{1}{n}$.

unweighted multigraphs). Formally, we show that if Bw bits can be transmitted over every edge of weight $w \geq 1$, for every $\lambda \geq 1$ and every $\alpha \geq 1$, there are λ-edge-connected networks with diameter $O(\log n)$ on which computing an α-approximate minimum cut requires time at least $\Omega\big(\sqrt{n/(B \log n)}\big)$. Further, for unweighted simple graphs, we show that computing an α-approximate minimum cut in λ-edge-connected networks of diameter $O(\log n)$ requires at least time $\Omega\Big(\frac{\sqrt{n/(B \log n)}}{(\alpha\lambda)^{1/4}}\Big)$.

In addition, our technique yields a structural result about λ-edge-connected graphs with small diameter. We show that for every $\lambda > 1$, there are λ-edge-connected graphs G with diameter $O(\log n)$ such that for any partition of the edges of G into spanning subgraphs, all but $O(\log n)$ of the spanning subgraphs have diameter $\Omega(n)$ (in the case of unweighted multigraphs) or $\Omega(n/\lambda)$ (in the case of unweighted simple graphs). As a corollary, we also get that when sampling each edge of such a graph with probability $p \leq \gamma/\log n$ for a sufficiently small constant $\gamma > 0$, with at least a positive constant probability, the subgraph induced by the sampled edges has diameter $\Omega(n)$ (in the case of unweighted multigraphs) and $\Omega(n/\lambda)$ (in the case of unweighted simple graphs). For lack of space, the details about these results are deferred to the full version [10].

1.1 Related Work in the Centralized Setting

Starting in the 1950s [5, 8], the traditional approach to the minimum cut problem was to use max-flow algorithms (cf. [7] and [20, Section 1.3]). In the 1990s, three new approaches were introduced which go away from the flow-based method and provide faster algorithms: The first method, presented by Gabow [9], is based on a matroid characterization of the min-cut and it finds a min-cut in $O(m + \lambda^2 n \log \frac{n}{m})$ steps, for any unweighted (but possibly directed) graph with edge connectivity λ. The second approach applies to (possibly) weighted but undirected graphs and is based on repeatedly identifying and contracting edges outside a min-cut until a min-cut becomes apparent (e.g., [14, 20, 25]). The beautiful *random contraction algorithm* (RCA) of Karger [14] falls into this category. In the basic version of RCA, the following procedure is repeated $O(n^2 \log n)$ times: contract uniform random edges one by one until only two nodes remain. The edges between these two nodes correspond to a cut in the original graph, which is a min-cut with probability at least $1/O(n^2)$. Karger and Stein [20] also present a more efficient implementation of the same basic idea, leading to total running time of $O(n^2 \log^3 n)$. The third method, which again applies to (possibly) weighted but undirected graphs, is due to Karger [18] and is based on a "semiduality" between minimum cuts and maximum spanning tree packings. This third method leads to the best known centralized minimum-cut algorithm [19] with running time $O(m \log^3 n)$.

For the approximation version of the problem (in undirected graphs), the main known results are as follows. Matula [24] presents an algorithm that finds a $(2 + \varepsilon)$-minimum cut for any constant $\varepsilon > 0$ in time $O((m + n)/\varepsilon)$. This algorithm is based on a graph search procedure called *maximum adjacency search*. Based on a modified version of the random contraction algorithm, Karger [17] presents an algorithm that finds a $(1 + \varepsilon)$-minimum cut in time $O(m + n \log^3 n/\varepsilon^4)$.

2 Preliminaries

Notations and Definitions: We usually work with an undirected weighted graph $G = (V, E, w)$, where V is a set of n vertices, E is a set of (undirected) edges $e = \{v, u\}$ for $u, v \in V$, and $w : E \to \mathbb{R}^+$ is a mapping from edges E to positive real numbers. For each edge $e \in E$, $w(e)$ denotes the weight of edge e. In the special case of unweighted graphs, we simply assume $w(e) = 1$ for each edge $e \in E$.

For a given non-empty proper subset $C \subset V$, we define the cut $(C, V \setminus C)$ as the set of edges in E with exactly one endpoint in set C. The size of this cut, denoted by $w(C)$ is the sum of the weights of the edges in set $(C, V \setminus C)$. The edge-connectivity $\lambda(G)$ of the graph is defined as the minimum size of $w(C)$ as C ranges over all nonempty proper subsets of V. A cut $(C, V \setminus C)$ is called *α-minimum*, for an $\alpha \geq 1$, if $w(C) \leq \alpha\lambda(G)$. When clear from the context, we sometimes use λ to refer to $\lambda(G)$.

Communicaton Model and Problem Statements: We use a standard *message passing model* (a.k.a. the CONGEST model [26]), where the execution proceeds in synchronous rounds and in each round, each node can send a message of size B bits to each of its neighbors. A typically standard case is $B = \Theta(\log n)$.

For upper bounds, for simplicity we assume that $B = \Theta(\log n)$[2]. For upper bounds, we further assume that B is large enough so that a constant number of node identifiers and edge weights can be packed into a single message. For $B = \Theta(\log n)$, this implies that each edge weight $w(e)$ is at most (and at least) polynomial in n. W.l.o.g., we further assume that edge weights are normalized and each edge weight is an integer in range $\{1, \ldots, n^{\Theta(1)}\}$. Thus, we can also view a weighted graph as a multi-graph in which all edges have unit weight and multiplicity at most $n^{\Theta(1)}$ (but still only $O(\log n)$ bits can be transmitted over all these parallel edges together).

For lower bounds, we assume a weaker model where $B \cdot w(e)$ bits can be sent in each round over each edge e. To ensure that at least B bits can be transmitted over each edge, we assume that the weights are scaled such that $w(e) \geq 1$ for all edges. For integer weights, this is equivalent to assuming that the network graph is an unweighted multigraph where each edge e corresponds to $w(e)$ parallel unit-weight edges.

In the problem of computing an *α-approximation of the minimum cut*, the goal is to find a cut $(C^*, V \setminus C^*)$ that is α-minimum. To indicate this cut in the distributed setting, each node v should know whether $v \in C^*$. In the problem of *α-approximation of the edge-connectivity*, all nodes must output an estimate $\tilde{\lambda}$ of λ such that $\tilde{\lambda} \in [\lambda, \lambda\alpha]$. In randomized algorithms for these problems, time complexities are fixed deterministically and the correctness guarantees are required to hold with high probability.

2.1 Black-Box Algorithms

In this paper, we make frequent use of a *connected component identification* algorithm due to Thurimella [29], which itself builds on the minimum spanning tree algorithm of Kutten and Peleg [22]. Given a graph $G(V, E)$ and a subgraph $H = (V, E')$ such that

[2] Note that by choosing $B = b \log n$ for some $b \geq 1$, in all our upper bounds, the term that does not depend on D could be improved by a factor $\sqrt{b}$.

$E' \subseteq E$, Thurimella's algorithm identifies the connected components of H by assigning a label $\ell(v)$ to each node $v \in V$ such that two nodes get the same label iff they are in the same connected component of H. The time complexity of the algorithm is $O(D + \sqrt{n}\log^* n)$ rounds, where D is the (unweighted) diameter of G. Moreover, it is easy to see that the algorithm can be made to produce labels $\ell(v)$ such that $\ell(v)$ is equal to the smallest (or the largest) id in the connected component of H that contains v. Furthermore, the connected component identification algorithm can also be used to test whether the graph H is connected (assuming that G is connected). H is not connected if and only if there is an edge $\{u, v\} \in E$ such that $\ell(u) \neq \ell(v)$. If some node u detects that for some neighbor v (in G), $\ell(u) \neq \ell(v)$, u broadcasts *not connected*. Connectivity of H can therefore be tested in D additional rounds. We refer to this as Thurimella's *connectivity-tester* algorithm. Finally, we remark that the same algorithms can also be used to solve k independent instances of the connected component identification problem or k independent instances of the connectivity-testing problem in $O(D + k\sqrt{n}\log^* n)$ rounds. This is achieved by pipelining the messages of the broadcast parts of different instances.

3 Edge Sampling and the Random Layering Technique

Here, we study the process of random edge-sampling and present a simple technique, which we call *random layering*, for analyzing the connectivity of the graph obtained through sampling. This technique also forms the basis of our min-cut approximation algorithm presented in the next section.

Edge Sampling: Consider an arbitrary unweighted multigraph $G = (V, E)$. Given a probability $p \in [0, 1]$, we define an *edge sampling experiment* as follows: choose subset $S \subseteq E$ by including each edge $e \in E$ in set S independently with probability p. We call the graph $G' = (V, S)$ *the sampled subgraph*.

We use the *random layering technique* to answer the following *network reliability* question: "How large should p be, as a function of minimum-cut size λ, so that the sampled graph is connected w.h.p.?"[3] Considering just one cut of size λ we see that if $p \leq \frac{1}{\lambda}$, then the probability that the sampled subgraph is connected is at most $\frac{1}{e}$. We show that $p \geq \frac{20 \log n}{\lambda}$ suffices so that the sampled subgraph is connected w.h.p. Note that this is non-trivial as a graph has exponential many cuts. It is easy to see that this bound is asymptotically optimal [23].

Theorem 1. *Consider an arbitrary unweighted multigraph $G = (V, E)$ with edge connectivity λ and choose subset $S \subseteq E$ by including each edge $e \in E$ in set S independently with probability p. If $p \geq \frac{20 \log n}{\lambda}$, then the sampled subgraph $G' = (V, S)$ is connected with probability at least $1 - \frac{1}{n}$.*

We remark that this result was known prior to this paper, via two different proofs by Lomonosov and Polesskii [23] and Karger [15]. The Lomonosov-Polesskii proof [23] uses an interesting coupling argument and shows that among the graphs of a given

[3] A rephrased version is, how large should the edge-connectivity λ of a network be such that it remains connected w.h.p. if each edge fails with probability $1 - p$.

edge-connectivity λ, a cycle of length n with edges of multiplicity $\lambda/2$ has the smallest probability of remaining connected under random sampling. Karger's proof [15] uses the powerful fact that the number of α-minimum cuts is at most $O(n^{2\alpha})$ and then uses basic probability concentration arguments (Chernoff and union bounds) to show that, w.h.p., each cut has at least one sampled edge. There are many known proofs for the $O(n^{2\alpha})$ upper bound on the number of α-minimum cuts (see [19]); an elegant argument follows from Karger's *random contraction algorithm* [14].

Our proof of Theorem 1 is simple and self-contained, and it is the only one of the three approaches that extends to the case of random vertex failures[4] [2, Theorem 1.5].

Proof (Proof of Theorem 1). Let $L = 20 \log n$. For each edge $e \in E$, we independently choose a uniform random *layer number* from the set $\{1, 2, \ldots, L\}$. Intuitively, we add the sampled edges layer by layer and show that with the addition of the sampled edges of each layer, the number of connected components goes down by at least a constant factor, with at least a constant probability, and independently of the previous layers. After $L = \Theta(\log n)$ layers, connectivity is achieved w.h.p.

We start by presenting some notations. For each $i \in \{1, \ldots, L\}$, let S_i be the set of sampled edges with layer number i and let $S_{i-} = \bigcup_{j=1}^{i} S_j$, i.e., the set of all sampled edges in layers $\{1, \ldots, i\}$. Let $G_i = (V, S_{i-})$ and let M_i be the number of connected components of graph G_i. We show that $M_L = 1$, w.h.p.

For any $i \in [1, L-1]$, since $S_{i-} \subseteq S_{(i+1)-}$, we have $M_{i+1} \leq M_i$. Consider the indicator variable X_i such that $X_i = 1$ iff $M_{i+1} \leq 0.87 M_i$ or $M_i = 1$. We show the following claim, after which, applying a Chernoff bound completes the proof.

Claim. For all $i \in [1, L-1]$ and $T \subseteq E$, we have $\Pr[X_i = 1 | S_{i-} = T] \geq 1/2$.

To prove this claim, we use *the principle of deferred decisions* [21] to view the two random processes of sampling edges and layering them. More specifically, we consider the following process: first, each edge is sampled and given layer number 1 with probability p/L. Then, each remaining edge is sampled and given layer number 2 with probability $\frac{p/L}{1-p/L} \geq p/L$. Similarly, after determining the sampled edges of layers 1 to i, each remaining edge is sampled and given layer number $i+1$ with probability $\frac{p/L}{1-(i\,p)/L} \geq p/L$. After doing this for L layers, any remaining edge is considered not sampled and it receives a random layer number from $\{1, 2, \ldots, L\}$. It is easy to see that in this process, each edge is independently sampled with probability exactly p and each edge e gets a uniform random layer number from $\{1, 2, \ldots, L\}$, chosen independently of the other edges and also independently of whether e is sampled or not.

Fix a layer $i \in [1, \ldots, L-1]$ and a subset $T \subseteq E$. Let $S_{i-} = T$ and consider graph $G_i = (V, S_{i-})$. Figure 1 presents an example graph G_i and its connected components. If $M_i = 1$ meaning that G_i is connected, then $X_i = 1$. Otherwise, suppose that $M_i \geq 2$. For each component $\mathcal{C}$ of G_i, call the component *bad* if $(\mathcal{C}, V \setminus \mathcal{C}) \cap S_{i+1} = \emptyset$. That is, $\mathcal{C}$ is bad if after adding the sampled edges of layer $i+1$, $\mathcal{C}$ does not get connected to any other component. We show that $\Pr[\mathcal{C} \textit{ is bad}] \leq \frac{1}{e}$.

[4] There, the question is, how large the vertex sampling probability p has to be chosen, as a function of vertex connectivity k, so that the vertex-sampled graph is connected, w.h.p. The extension to the vertex version requires important modifications and leads to $p = \Omega(\frac{\log n}{\sqrt{k}})$ being a sufficient condition. Refer to [2, Section 3] for details.

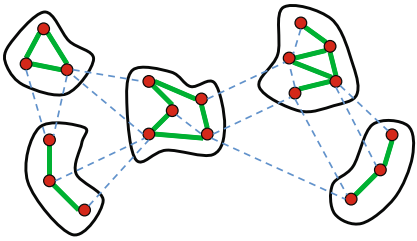

Fig. 1. Graph G_i and its connected components. The green solid links represent edges in S_{i-} and the blue dashed links represent $E \setminus S_{i-}$.

Since G is λ-edge connected, we have $w(C) \geq \lambda$. Moreover, none of the edges in $(\mathcal{C}, V \setminus \mathcal{C})$ is in S_{i-}. Thus, using the principle of deferred decisions as described, each of the edges of the cut $(\mathcal{C}, V \setminus \mathcal{C})$ has probability $\frac{p/L}{1-(ip)/L} \geq p/L$ to be sampled and given layer number $i+1$, i.e., to be in S_{i+1}. Since $p \geq \frac{20 \log n}{\lambda}$, the probability that none of the edges $(\mathcal{C}, V \setminus \mathcal{C})$ is in set S_{i+1} is at most $(1-p/L)^{\lambda} \leq 1/e$. Thus, $\Pr[\mathcal{C} \textit{ is bad}] \leq 1/e$. Having this, since each component that is not bad gets connected to at least one other component (when we look at graph G_{i+1}), a simple application of Markov's inequality proves the claim, and after that, a Chernoff bound completes the proof. See [10] for details. □

Theorem 1 provides a very simple approach for finding an $O(\log n)$-approximation of the edge connectivity of a network graph G in $O(D + \sqrt{n} \log^2 n \log^* n)$ rounds, simply by trying exponentially growing sampling probabilities and checking the connectivity. The proof appears the full version [10]. We note that a similar basic approach has been used to approximate *the size* of min-cut in the streaming model [1].

Corollary 1. *There exists a distributed algorithm that for any unweighted multi-graph $G = (V, E)$, in $O(D + \sqrt{n} \log^2 n \log^* n)$ rounds, finds an approximation $\tilde{\lambda}$ of the edge connectivity such that $\tilde{\lambda} \in [\lambda, \lambda \cdot \Theta(\log n)]$ with high probability.*

4 Min-Cut Approximation by Random Layering

Now we use random layering to design a min-cut approximation algorithm. We present the outline of the algorithm and its major ideas but defer putting the pieces together to the proof of Theorem 2 in the full version [10].

Theorem 2. *There is a distributed algorithm that, for any $\epsilon \in (0, 1)$, finds an $O(\epsilon^{-1})$-minimum cut in $O(D) + O(n^{0.5+\epsilon} \log^3 n \log \log n \log^* n)$ rounds, w.h.p.*

4.1 Algorithm Outline

The algorithm is based on closely studying the sampled graph when the edge-sampling probability is between the two extremes of $\frac{1}{\lambda}$ and $\frac{\Theta(\log n)}{\lambda}$. Throughout this process, we identify a set $\mathcal{F}$ of $O(n \log n)$ cuts such that, with at least a 'reasonably large probability', $\mathcal{F}$ contains at least one 'small' cut.

The Crux of the Algorithm: Sample edges with probability $p = \frac{\epsilon \log n}{2\lambda}$ for a small $\epsilon \in (0,1)$. Also, assign each edge to a random layer in $[1, \ldots, L]$, where $L = 20 \log n$. For each layer $i \in [1, \ldots, L-1]$, let S_i be the set of sampled edges of layer i and let $S_{i-} = \bigcup_{j=1}^{i} S_j$. For each layer $i \in [1, \ldots, L-1]$, for each component $\mathcal{C}$ of graph $G_i = (V, S_{i-})$, add the cut $(\mathcal{C}, V \setminus \mathcal{C})$ to the collection $\mathcal{F}$.

We show that with probability at least $n^{-\epsilon}/2$, at least one of the cuts in $\mathcal{F}$ is an $O(\epsilon^{-1})$-minimum cut. Note that thus repeating the experiment for $\Theta(n^{\epsilon} \log n)$ times is enough to get that an $O(\epsilon^{-1})$-minimum cut is found w.h.p.

Theorem 3. *Consider performing the above sampling and layering experiment with edge sampling probability $p = \frac{\epsilon \log n}{2\lambda}$ for $\epsilon \in (0,1)$ and $L = 20 \log n$ layers. Then,* $\Pr[\mathcal{F}$ *contains an* $O(\epsilon^{-1})$*-minimum cut*$] \geq n^{-\epsilon}/2$.

Proof. Fix an edge sampling probability $p = \frac{\epsilon \log n}{2\lambda}$ for an $\epsilon \in (0,1)$ and let $\alpha = 40\epsilon^{-1}$. We say that a sampling and layering experiment is *successful* if $\mathcal{F}$ contains an α-minimum cut or if the sampled graph $G_L = (V, S_{L-})$ is connected. We first show that each experiment is *successful* with probability at least $1 - \frac{1}{n}$. The proof of this part is very similar to that of Theorem 1.

For an arbitrary layer number $1 \leq i \leq L-1$, consider graph $G_i = (V, S_{i-})$. If $M_i = 1$ meaning that G_i is connected, then G_L is also connected. Thus, in that case, the experiment is successful and we are done. In the more interesting case, suppose $M_i \geq 2$. For each component $\mathcal{C}$ of G_i, consider the cut $(\mathcal{C}, V \setminus \mathcal{C})$. If any of these cuts is α-minimum, then the experiment is successful as then, set $\mathcal{F}$ contains an α-minimum cut. On the other hand, suppose that for each component $\mathcal{C}$ of G_i, we have $w(C) \geq \alpha\lambda$. Then, for each such component $\mathcal{C}$, each of the edges of cut $(\mathcal{C}, V \setminus \mathcal{C})$ has probability $\frac{p/L}{1-(i\,p)/L} \geq p/L$ to be in set S_{i+1} and since $w(\mathcal{C}) \geq \alpha\lambda$, where $\alpha = 20\epsilon^{-1}$, the probability that none of the edges of this cut in set S_{i+1} is at most $(1-p/L)^{\alpha\lambda} \leq e^{\frac{p}{L}\cdot\alpha\lambda} = e^{-\frac{\epsilon \log n}{2\lambda}\cdot\frac{1}{L}\cdot\frac{40}{\epsilon}\cdot\lambda} = 1/e$. Hence, the probability that component $\mathcal{C}$ is *bad* as defined in the proof of Theorem 1 (i.e., in graph G_{i+1}, it does not get connected to any other component) is at most $1/e$. The rest of the proof can be completed exactly as the last paragraph of of the proof of Theorem 1, to show that

$$\Pr[\textit{successful experiment}] \geq 1 - 1/n.$$

Using a union bound, we know that

$$\Pr[\textit{successful experiment}] \leq \Pr[\mathcal{F} \textit{ contains an } \alpha\textit{-min cut}] + \Pr[G_L \textit{ is connected}].$$

On the other hand,

$$\Pr[G_L \textit{ is connected}] \leq 1 - n^{-\epsilon}.$$

This is because, considering a single mininmum cut of size λ, the probability that none of the edges of this cut are sampled, in which case the sampled subgraph is disconnected, is $(1 - \frac{\epsilon \log n}{2\lambda})^{\lambda} \geq n^{-\epsilon}$. Hence, we can conclude that

$$\Pr[\mathcal{F} \textit{ contains an } \alpha\textit{-min cut}] \geq (1 - 1/n) - (1 - n^{-\epsilon}) = n^{-\epsilon} - 1/n \geq n^{-\epsilon}/2. \quad \square$$

Remark: It was brought to our attention that the approach of Theorem 3 bears some cosmetic resemblance to the technique of Goel, Kapralov and Khanna [11]. As noted by Kapralov [13], the approaches are fundamentally different; the only similarity is having $O(\log n)$ repetitions of sampling. In [11], the objective is to estimate the *strong-connectivity* of edges via a streaming algorithm. See [11] for related definitions and note also that strong-connectivity is (significantly) different from (standard) connectivity. In a nutshell, [11] uses $O(\log n)$ iterations of sub-sampling, each time further sparsifying the graph until at the end, all edges with strong-connectivity less than a threshold are removed (and identified) while edges with strong connectivity that is a $\Theta(\log n)$ factor larger than the threshold are preserved (proven via Benczur-Karger's sparsification).

4.2 Testing Cuts

So far we know that $\mathcal{F}$ contains an α-minimum cut with a reasonable probability. We now need to devise a distributed algorithm to read or test the sizes of the cuts in $\mathcal{F}$ and find that α-minimum cut, in $O(D) + \tilde{O}(\sqrt{n})$ rounds.

Consider a layer i and the graph $G_i = (V, S_{i-})$. Notice that we do not need to read the exact size of the cut $(\mathcal{C}, V \setminus \mathcal{C})$. Instead, it is enough to devise a *test* that *passes* w.h.p. if $w(C) \leq \alpha\lambda$, and *does not pass* w.h.p. if $w(C) \geq (1+\delta)\alpha\lambda$, for a small constant $\delta \in (0, 1/4)$. In the distributed realization of such a test, it would be enough if all the nodes in $\mathcal{C}$ consistently know whether the test passed or not. Next, we explain a simple algorithm for such a test. This test itself uses random edge sampling. Given such a test, in each layer $i \in [1, \ldots, L-1]$, we can test all the cuts and if any cut passes the test, meaning that, w.h.p., it is a $((1+\delta)\alpha)$-minimum cut, then we can pick such a cut.[5]

Lemma 1. *Given a subgraph $G' = (V, E')$ of the network graph $G = (V, E)$, a threshold κ and $\delta \in (0, 1/4)$, there exists a randomized distributed cut-tester algorithm with round complexity $\Theta\big(D + \frac{1}{\delta^2}\sqrt{n}\log n \log^* n\big)$ such that, w.h.p., for each node $v \in V$, we have: Let $\mathcal{C}$ be the connected component of G' that contains v. If $w(\mathcal{C}) \leq \kappa/(1+\delta)$, the test passes at v, whereas if $w(\mathcal{C}) \geq \kappa(1+\delta)$, the test does not pass at v.*

For pseudo-code, we refer to the full version [10]. We first run Thurimella's connected component identification algorithm (refer to Section 2.1) on graph G for subgraph G', so that each node $v \in V$ knows the smallest id in its connected component of graph G'. Then, each node v adopts this label $componentID$ as its own id (temporarily). Thus, nodes of each connected component of G' will have the same id. Now, the test runs in $\Theta(\log^2 n/\delta^2)$ *experiments*, each as follows: in the j^{th} experiment, for each edge $e \in E \setminus E'$, put edge e in set E_j with probability $p' = 1 - 2^{-\frac{1}{\kappa}}$. Then, run Thurimella's algorithm on graph G with subgraph $H_j = (V, E' \cup E_j)$ and with the new ids twice, such that at the end, each node v knows the smallest and the largest id in its connected component of H_j. Call these new labels $\ell_j^{min}(v)$ and $\ell_j^{max}(v)$, respectively. For a node v of a component $\mathcal{C}$ of G_i, we have that $\ell_j^{min}(v) \neq v.id$ or $\ell_j^{max}(v) \neq v.id$ iff at least one of the edges of cut $(\mathcal{C}, V \setminus \mathcal{C})$ is sampled in E_j, i.e., $(\mathcal{C}, V \setminus \mathcal{C}) \cap E_j \neq \emptyset$. Thus, each node v of each component $\mathcal{C}$ knows whether $(\mathcal{C}, V \setminus \mathcal{C}) \cap E_j \neq \emptyset$ or not.

[5] This can be done for example by picking the cut which passed the test and for which the related component has the smallest id among all the cuts that passed the test.

Moreover, this knowledge is consistent between all the nodes of component $\mathcal{C}$. After $\Theta(\log n/\delta^2)$ experiments, each node v of component $\mathcal{C}$ considers the test *passed* iff v noticed $(\mathcal{C}, V\setminus\mathcal{C})\cap E_j \neq \emptyset$ in at most half of the experiments. We defer the calculations of the proof of Lemma 1 to of the full version [10].

5 Min-Cut Approximation via Matula's Approach

In [24], Matula presents an elegant centralized algorithm that for any constant $\varepsilon > 0$, finds a $(2+\varepsilon)$-min-cut in $O(|V|+|E|)$ steps. Here, we explain how with the help of a few additional elements, this general approach can be used in the distributed setting, to find a $(2+\varepsilon)$-minimum cut in $O\big((D+\sqrt{n}\log^* n)\log^2 n \log\log n \cdot \frac{1}{\varepsilon^5}\big)$ rounds. We first recap the concept of *sparse certificates for edge connectivity*.

Definition 1. *For a given unweighted multi-graph $H=(V_H,E_H)$ and a value $k>0$, a set $E^*\subseteq E_H$ of edges is a* sparse certificate for k-edge-connectivity *of H if (1) $|E^*|\leq k|V_H|$, and (2) for each edge $e\in E_H$, if there exists a cut $(\mathcal{C}, V\setminus\mathcal{C})$ of H such that $|(\mathcal{C})|\leq k$ and $e\in(\mathcal{C}, V\setminus\mathcal{C})$, then we have $e\in E^*$.*

Thurimella [29] presents a simple distributed algorithm that finds a sparse certificate for k-edge-connectivity of a network graph G in $O(k(D+\sqrt{n}\log^* n))$ rounds. With simple modifications, we get a generalized version, presented in Lemma 2. Details of these modification appear in the full version of this paper [10].

Lemma 2. *Let E_c be a subset of the edges of the network graph G and define the virtual graph $G'=(V',E')$ as the multi-graph that is obtained by contracting all the edges of G that are in E_c. Using the modified version of Thurimella's certificate algorithm, we can find a set $E^*\subseteq E\setminus E_c$ that is a sparse certificate for k-edge-connectivity of G', in $O(k(D+\sqrt{n}\log^* n))$ rounds.*

Following the approach of Matula's centralized algorithm[6] [24], and with the help of the sparse certificate algorithm of Lemma 2 and the random sparsification technique of Karger [15], we get the following result.

Theorem 4. *There is a distributed algorithm that, for any constant $\varepsilon > 0$, finds a $(2+\varepsilon)$-minimum cut in $O((D+\sqrt{n}\log^* n)\log^2 n\log\log n\cdot\frac{1}{\varepsilon^5})$ rounds.*

Proof (Proof Sketch). We assume that nodes know a $(1+\varepsilon/10)$-factor approximation $\tilde{\lambda}$ of the edge connectivity λ, and explain a distributed algorithm with round complexity $O((D+\sqrt{n}\log^* n)\log^2 n\cdot\frac{1}{\varepsilon^4})$. Note that this assumption can be removed at the cost of a $\Theta(\frac{\log\log n}{\log(1+\varepsilon/10)})=\Theta(\log\log n\cdot\frac{1}{\varepsilon})$ factor increase in round complexity by trying $\Theta(\frac{\log\log n}{\varepsilon})$ exponential guesses $\tilde{\lambda}(1+\varepsilon/10)^i$ for $i\in[0,\Theta(\frac{\log\log n}{\varepsilon})]$ where $\tilde{\lambda}$ is an $O(\log n)$-approximation of the edge-connectivity, which can be found by Corollary 1.

For simplicity, we first explain an algorithm that finds a $(2+\varepsilon)$-minimum cut in $O(\lambda(D+\sqrt{n}\log^* n)\log n\cdot\frac{1}{\varepsilon^2})$ rounds. Then, we explain how to reduce the round complexity to $O((D+\sqrt{n}\log^* n)\log^2 n\cdot\frac{1}{\varepsilon^4})$.

[6] We remark that Matula [24] never uses the name *sparse certificate* but he performs *maximum adjacency search* which indeed generates a sparse certificate.

Pseudo-code is given in the full version [10]. First, we compute a sparse certificate E^* for $\tilde{\lambda}(1+\varepsilon/5)$-edge-connectivity of G, using Thurimella's algorithm. Now consider the graph $H = (V, E \setminus E^*)$. We have two cases: either (a) H has at most $|V|(1-\varepsilon/10)$ connected components, or (b) there is a connected component $\mathcal{C}$ of H such that $w(\mathcal{C}) \leq \frac{2\tilde{\lambda}(1+\varepsilon/10)(1+\varepsilon/5)}{1-\varepsilon/10} \leq (2+\varepsilon)\lambda$. Note that if (a) does not hold, case (b) follows because H has at most $(1+\varepsilon/5)\tilde{\lambda}|V|$ edges.

In Case (b), we can find a $(2+\varepsilon)$-minimum cut by testing the connected components of H versus threshold $\kappa = \tilde{\lambda}(2+\varepsilon/3)$, using the Cut-Tester algorithm presented in Lemma 1. In Case (a), we solve the problem recursively on the virtual graph $G' = (V', E')$ that is obtained by contracting all the edges of G that are in $E_c = E \setminus E^*$. Note that this contraction preserves all the cuts of size at most $\tilde{\lambda}(1+\varepsilon/5) \geq \lambda$ but reduces the number of nodes (in the virtual graph) at least by a $(1-\varepsilon/10)$-factor. Consequently, $O(\log(n)/\varepsilon)$ recursions reduce the number of components to at most 2 while preserving the min-cut.

The dependence on λ can be removed by considering the graph $G_S = (V, E_S)$, where E_S independently contains every edge of G with probability $\Theta\big(\frac{\log n}{\varepsilon^2 \lambda}\big)$. It can be shown that the edge connectivity of G_S is $\Theta(\log(n)/\varepsilon^2)$ and a min-cut of G_S gives a $(1+O(\varepsilon))$-min-cut of G. The details appear in the full version [10]. □

6 Lower Bounds

In this section, we describe a lower bound that allows to strengthen and generalize some of the lower bounds of Das Sarma et al. from [4]. Our lower bound uses the same basic approach as the lower bounds in [4]. The lower bounds of [4] are based on an n-node graph G with diameter $O(\log n)$ and two distinct nodes s and r. The proof deals with distributed protocols where node s gets a b-bit input x, node r gets a b-bit input y, and apart from x and y, the initial states of all nodes are globally known. Slightly simplified, the main technical result of [4] (Simulation Theorem 3.1) states that if there is a randomized distributed protocol that correctly computes the value $f(x,y)$ of a binary function $f : \{0,1\}^b \times \{0,1\}^b \to \{0,1\}$ with probability at least $1-\varepsilon$ in time T (for sufficiently small T), then there is also a randomized ε-error two-party protocol for computing $f(x,y)$ with communication complexity $O(TB\log n)$. For our lower bounds, we need to extend the simulation theorem of [4] to a larger family of networks and to a slightly larger class of problems.

6.1 Generalized Simulation Theorem

Distributed Protocols: Given a weighted network graph $G = (V, E, w)$ ($\forall e \in E : w(e) \geq 1$), we consider distributed tasks for which each node $v \in V$ gets some private input $x(v)$ and every node $v \in V$ has to compute an output $y(v)$ such that the collection of inputs and outputs satisfies some given specification. To solve a given distributed task, the nodes of G apply a distributed protocol. We assume that initially, each node $v \in V$ knows its private input $x(v)$, as well as the set of neighbors in G. Time is divided into synchronous rounds and in each round, every node can send at most $B \cdot w(e)$ bits

over each of its incident edges e. We say that a given (randomized) distributed protocol solves a given distributed task with error probability ε if the computed outputs satisfy the specification of the task with probability at least $1-\varepsilon$.

Graph Family $\mathcal{G}(n,k,c)$: For parameters n, k, and c, we define the *family of graphs* $\mathcal{G}(n,k,c)$ as follows. A weighted graph $G=(V,E,w)$ is in the family $\mathcal{G}(n,k,c)$ iff $V=\{1,\dots,n\}$ and for all $h\in\{1,\dots,n\}$, the total weight of edges between nodes in $\{1,\dots,h\}$ and nodes in $\{h+k+1,\dots,n\}$ is at most c. We consider distributed protocols on graphs $G\in\mathcal{G}(n,k,c)$ for given n, k, and c. For an integer $\eta\geq 1$, we define $L_\eta:=\{1,\dots,\eta\}$ and $R_\eta:=\{n-\eta+1,\dots,n\}$. Given a parameter $\eta\geq 1$ and a network $G\in\mathcal{G}(n,k,c)$, we say that a two-party protocol between Alice and Bob η-solves a given distributed task for G with error probability ε if a) initially Alice knows all inputs and all initial states of nodes in $V\setminus R_\eta$ and Bob knows all inputs and all initial states of nodes in $V\setminus L_\eta$, and b) in the end, Alice outputs $y(v)$ for all $v\in L_{n/2}$ and Bob outputs $y(v)$ for all $v\in R_{n/2}$ such that with probability at least $1-\varepsilon$, all these $y(v)$ are consistent with the specification of the given distributed task. A two-party protocol is said to be *public coin* if Alice and Bob have access to a common random string. The proof of the following theorem appears in the full version [10].

Theorem 5 (Generalized Simulation Theorem). *Assume we are given positive integers n, k, and η, a parameter $c\geq 1$, as well as a subfamily $\tilde{\mathcal{G}}\subseteq\mathcal{G}(n,k,c)$. Further assume that for a given distributed task and graphs $G\in\tilde{\mathcal{G}}$, there is a randomized protocol with error probability ε that runs in $T\leq(n-2\eta)/(2k)$ rounds. Then, there exists a public-coin two-party protocol that η-solves the given distributed task on graphs $G\in\tilde{\mathcal{G}}$ with error probability ε and communication complexity at most $2BcT$.*

We now describe a generic construction to obtain graphs of the family $\mathcal{G}(n,k,c)$. Given some integer $n>0$, we define $T_n=(V,E_T)$ to be a fixed unweighted binary tree on the nodes $V=\{1,\dots,n\}$ with depth $\lceil\log_2 n\rceil$ where an *in-order DFS traversal* of T_n (starting at the root) reproduces the natural order $1,2,\dots,n$. The tree T_n can thus be seen as a binary search tree: Given any node i, for all nodes j of the left subtree of i, it holds that $j<i$ and for all nodes j of the right subtree of i, it holds that $j>i$.

Lemma 3. *Given an integer $p\in\{1,\dots,n-1\}$, consider the cut $(S_p,V\setminus S_p)$, where $S_p=\{1,\dots,p\}$. For every $p\in\{1,\dots,n-1\}$, the number of edges between over the cut $(S_p,V\setminus T_p)$ is at most $\lceil\log_2 n\rceil$.*

Using the tree T_n, we can construct graphs from the family $\mathcal{G}(n,k,c)$ for $c=\lceil\log_2 n\rceil$. Let $\mathcal{H}(n,k)$ be the family of weighted graphs $H=(V,E_H,w_H)$ with node set $V=\{1,\dots,n\}$ such that for all edges $\{i,j\}\in E_H$, $|j-i|\leq k$. Given a graph $H\in\mathcal{H}(n,k)$, we define a graph $G(H)=(V,E,w)$ with node set $V=\{1,\dots,n\}$ as follows: (a) The edge set E of $G(H)$ is $E:=E_H\cup E_T$. (b) The weight $w(e)$ of an edge $e\in E$ is given as $w(e):=\max\{1,w_H(e)\}$.

Lemma 4. *Given a graph $H\in\mathcal{H}(n,k)$, graph $G(H)\in\mathcal{G}(n,k,c)$ for $c=\lceil\log_2 n\rceil$. Further, the diameter of $G(H)$ is $O(\log n)$.*

6.2 Lower Bound for Approximating Minimum Cut

We start by proving a lower bound on approximating min-cut in weighted graphs (or equivalently in unweighted multigraphs).

Theorem 6. *In weighted graphs, for any $\alpha \geq 1$ and any $\lambda \geq 1$, computing an α-approximate minimum cut requires at least $\Omega\big(D + \sqrt{n/(B \log n)}\big)$ rounds.*

Proof. We prove the theorem by reducing from the two-party set disjointness problem [3,12,28]. Assume that as input, Alice gets a set X and Bob get a set Y such that both X and Y are of size p and the elements of X and Y are from a universe of size $O(p)$. It is known that for Alice and Bob need to exchange at least $\Omega(p)$ bits to determine whether X and Y are disjoint [12, 28]. This lower bound holds even for public coin randomized protocols with constant error probability and it also holds if Alice and Bob are given the promise that if X and Y intersect, they intersect in exactly one element [28]. As a consequence, if Alice and Bob receive sets X and Y of size p as inputs such that $|X \cap Y| = 1$, finding $X \cap Y$ also requires Alice and Bob to exchange $\Omega(p)$ bits.

Assume that there is a protocol to find an α-minimum cut or to α-approximate the size of a minimum cut in time T with a constant error probability ε. In both cases, we show that Alice and Bob can use this protocol to efficiently solve set disjointness by simulating the distributed protocol on a special network.

We now describe the construction of this network. Let a and b be two positive integer parameters. We construct a graph $G \in \mathcal{G}(n, \lambda, O(\log n))$ as follows: First, we construct a weighted graph $H = (V_H, E_H, w_H) \in \mathcal{H}(a, 1)$ where the node set of H is $V_H = \{1, \ldots, a\}$ and there is an edge e of weight $w_H(e) = \alpha\lambda + 1$ between nodes i and j if and only if $|i-j| = 1$. By Lemma 4, we can then get a graph $G(H) \in \mathcal{G}(a, 1, O(\log n))$. To get a graph G, we add b additional copies of graph H. Call node i in the original copy $(i, 0)$ and node i in the j^{th} additional copy node (i, j). In each copy $j \geq 1$, we connect node $(1, j)$ with node $(1, 0)$ by an edge of weight λ. By renaming node (i, j) to $\kappa(i, j) := j + (i-1)(b+1)$, we can see that graph G is in $\mathcal{G}(a(b+1), b+1, O(\log n))$. In the following, let $n = a(b+1)$ be the number of nodes of G. The first $b+1$ nodes of G are nodes $(1, j)$ for $0 \leq j \leq b$, the last $b+1$ nodes of G are nodes (a, j) for $0 \leq j \leq b$. Note that graph G is exactly λ-edge connected as any of the edges $\{(1, j), (1, 0)\}$ defines a cut of size λ. Note also that every cut which divides one of the copies of H into two or more parts has size at least $\alpha\lambda + 1$.

Assume that Alice and Bob need to solve a set disjointness instance where $X \subset \{1, \ldots, b\}$, $Y \subset \{1, \ldots, b\}$, $|X \cap Y| \leq 1$, and $|X|, |Y| = \Omega(b)$. The graph G is extended such that the minimum cut problem in G represents the given set cover instance. For each $x \notin X$, the weight of the edge $\{(1, x), (1, 0)\}$ is increased to $\alpha\lambda + 1$. Further, for every $y \notin Y$, we add an edge $\{(a, y), (a, 0)\}$ of weight $\alpha\lambda + 1$. Now, if and only if $X \cap Y = \emptyset$, every copy of H is connected to the first copy by a link of weight $\alpha\lambda + 1$. Therefore, if X and Y are disjoint, the size of a minimum cut is at least $\alpha\lambda + 1$ and if X and Y intersect, there is a cut of size λ.

Alice knows the initial states of nodes (i, j) for all $i < a$ and thus for the nodes (i, j) with $1 \leq \kappa(i, j) < n - b$ (i.e., all except the last $b + 1$ nodes) and Bob knows the initual states of nodes (i, j) for all $i > 1$ and thus for the nodes (i, j) with $b + 1 < \kappa(i, j) \leq n$ (i.e., all except the first $b+1$ nodes). If we have $T < (n-2)/(2(b+1)) =$

$O(n/b) = O(a)$ for the time complexity T of the distributed minimum cut approximation protocol, Theorem 5 implies that Alice and Bob can $(b+1)$-solve the distributed task of α-approximating the minimum cut with total communication complexity at most $O(TB \log n)$. As a consequence, Alice and Bob can also solve the given set disjointness instance using the same protocol and from the known set disjointness communication complexity lower bound, we therefore get $TB \log n = \Omega(b)$. Choosing $a = \Theta(\sqrt{n/(B \log n)})$ and $b = \Theta(\sqrt{nB \log n})$ this implies the claimed lower bound for approximating the size of the minimum cut. Assuming that Alice and Bob already know that $|X \cap Y| = 1$, the communication complexity lower bound on finding $X \cap Y$ also implies the same lower bound for finding an α-minimum cut, even if the size λ of the minimum cut is known. □

We now present our lower bound about min-cut approximation in unweighted simple graphs.

Theorem 7. *In unweighted simple graphs, for any $\alpha \geq 1$ and $\lambda \geq 1$, computing an α-approximate minimum cut requires at least $\Omega\big(D + \sqrt{\frac{n}{B\sqrt{\alpha\lambda}\log n}}\big)$ rounds.*

Proof (Proof Sketch). The proof is essentially done in the same way as the proof of Theorem 6. We therefore only describe the differences between the proofs. Because in a simple unweighted graph, we cannot add edges with different weights and we cannot add multiple edges, we have to construct the graph differently. Let us first describe the simple, unweighted graph H' corresponding to H in the construction of Theorem 7. Instead of a being path of length a with edges of weight $\alpha\lambda + 1$, H' is a sequence of a cliques of size $\lceil\sqrt{\alpha\lambda + 1}\rceil$. Adjacent cliques are connected by complete bipartite graphs (with at least $\alpha\lambda + 1$ edges). We again have $b + 1$ copies of H', where copy 0 is augmented with a complete binary tree by using Lemma 4. Each edge $\{(1,0),(1,j)\}$ of weight λ is replaced by λ edges between clique $(1,0)$ and clique $(1,j)$. Edges of weight $\alpha\lambda + 1$ between nodes $(1,0)$ and $(1,j)$ and between nodes $(a,0)$ and (a,j) are replaced by complete bipartite graphs between the respective cliques. Again, by simulating a minimum cut approximation algorithm on the constructed graph, Alice and Bob can solve a given set disjointness instance for a universe of size b. However, the number of nodes of the network in this case is $\Theta(ab\sqrt{\alpha\lambda})$ leading to the lower bound claimed by the theorem. □

Acknowledgment. We thank David Karger for helpful discussions in the early stages of this work and thank Michael Kapralov for discussing cosmetic similarities with [11].

References

1. Ahn, K.J., Guha, S., McGregor, A.: Graph sketches: sparsification, spanners, and subgraphs. In: Proc. of the 31st Symp. on Princ. of Database Sys., PODS 2012, pp. 5–14 (2012)
2. Censor-Hillel, K., Ghaffari, M., Kuhn, F.: A new perspective on vertex connectivity. arXiv (2013), `http://arxiv.org/abs/1304.4553`
3. Chattapodhyay, A., Pitassi, T.: The story of set disjointness. SIGACT News Complexity Theory Column 67 (2011)

4. Das Sarma, A., Holzer, S., Kor, L., Korman, A., Nanongkai, D., Pandurangan, G., Peleg, D., Wattenhofer, R.: Distributed verification and hardness of distributed approximation. SIAM J. on Comp. 41(5), 1235–1265 (2012)
5. Elias, P., Feinstein, A., Shannon, C.E.: Note on maximum flow through a network. IRE Transactions on Information Theory IT-2, 117–199 (1956)
6. Elkin, M.: Distributed approximation: a survey. SIGACT News 35(4), 40–57 (2004)
7. Ford, L.R., Fulkerson, D.R.: Flows in Networks. Princeton Univ. Press (2010)
8. Ford, L.R., Fulkersonn, D.R.: Maximal flow through a network. Canad. J. Math. 8, 399–404 (1956)
9. Gabow, H.N.: A matroid approach to finding edge connectivity and packing arborescences. In: Proc. 23rd ACM Symposium on Theory of Computing (STOC), pp. 112–122 (1991)
10. Ghaffari, M., Kuhn, F.: Distributed minimum cut approximation. arXiv (2013), `http://arxiv.org/abs/1305.5520`
11. Goel, A., Kapralov, M., Khanna, S.: Graph sparsification via refinement sampling. arXiv (2010), `http://arxiv.org/abs/1004.4915`
12. Kalyanasundaram, B., Schnitger, G.: The probabilistic communication complexity of set intersection. SIAM J. Discrete Math. 5(4), 545–557 (1992)
13. Kapralov, M.: Personal communication (August 2013)
14. Karger, D.R.: Global min-cuts in $\mathcal{RNC}$, and other ramifications of a simple min-out algorithm. In: Prc. 4th ACM-SIAM Symp. on Disc. Alg. (SODA), pp. 21–30 (1993)
15. Karger, D.R.: Random sampling in cut, flow, and network design problems. In: Proc. 26th ACM Symposium on Theory of Computing (STOC), STOC 1994, pp. 648–657 (1994)
16. Karger, D.R.: Random sampling in cut, flow, and network design problems. In: Proc. 26th ACM Symposium on Theory of Computing (STOC), pp. 648–657 (1994)
17. Karger, D.R.: Using randomized sparsification to approximate minimum cuts. In: Proc. 5th ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 424–432 (1994)
18. Karger, D.R.: Minimum cuts in near-linear time. In: Proc. 28th ACM Symp. on Theory of Computing (STOC), pp. 56–63 (1996)
19. Karger, D.R.: Minimum cuts in near-linear time. J. ACM 47(1), 46–76 (2000)
20. Karger, D.R., Stein, C.: An $\tilde{O}(n^2)$ algorithm for minimum cuts. In: Proc. 25th ACM Symposium on Theory of Computing (STOC), pp. 757–765 (1993)
21. Knuth, D.E.: Stable Marriage and Its Relation to Other Combinatorial Problems: An Introduction to the Mathematical Analysis of Algorithms. AMS (1996)
22. Kutten, S., Peleg, D.: Fast distributed construction of k-dominating sets and applications. In: Proc. of the 14th Annual ACM Symp. on Principles of Dist. Comp., PODC 1995, pp. 238–251 (1995)
23. Lomonosov, M.V., Polesskii, V.P.: Lower bound of network reliability. Problems of Information Transmission 7, 118–123 (1971)
24. Matula, D.W.: A linear time $2 + \varepsilon$ approximation algorithm for edge connectivity. In: Proc. of the 4th Annual ACM-SIAM Symposium on Disc. Alg., SODA 1993, pp. 500–504 (1993)
25. Nagamochi, H., Ibaraki, T.: Computing edge-connectivity in multigraphs and capacitated graphs. SIAM J. Discret. Math. 5(1), 54–66 (1992)
26. Peleg, D.: Distributed Computing: A Locality-Sensitive Approach. SIAM (2000)
27. Picard, J.C., Queyranne, M.: Selected applications of minimum cuts in networks. Infor. 20, 19–39 (1982)
28. Razborov, A.A.: On the distributional complexity of disjointness. Theor. Comp. Sci. 106, 385–390 (1992)
29. Thurimella, R.: Sub-linear distributed algorithms for sparse certificates and biconnected components. Journal of Algorithms 23(1), 160–179 (1997)