On Streaming and Communication Complexity of the Set Cover Problem

Erik D. Demaine, Piotr Indyk, Sepideh Mahabadi, and Ali Vakilian

Massachusetts Instittute of Technology (MIT)
{edemaine,indyk,mahabadi,vakilian}@mit.edu

Abstract. We develop the first streaming algorithm and the first two-party communication protocol that uses a constant number of passes/rounds and sublinear space/communication for logarithmic approximation to the classic Set Cover problem. Specifically, for n elements and m sets, our algorithm/protocol achieves a space bound of $O(m \cdot n^{\delta} \log^2 n \log m)$ using $O(4^{1/\delta})$ passes/rounds while achieving an approximation factor of $O(4^{1/\delta} \log n)$ in polynomial time (for $\delta = \Omega(1/\log n)$). If we allow the algorithm/protocol to spend exponential time per pass/round, we achieve an approximation factor of $O(4^{1/\delta})$. Our approach uses randomization, which we show is necessary: no deterministic constant approximation is possible (even given exponential time) using $o(mn)$ space. These results are some of the first on streaming algorithms and efficient two-party communication protocols for approximation algorithms. Moreover, we show that our algorithm can be applied to multi-party communication model.

1 Introduction

The Set Cover problem is one of the classic tasks in combinatorial optimization. Given a set of n elements $\mathcal{E}$ and a collection of m sets $\mathcal{S} = \{S_1, \ldots, S_m\}$, the goal of the problem is to pick a subset $\mathcal{I} \subset \mathcal{S}$ such that (i) $\mathcal{I}$ *covers* $\mathcal{E}$, i.e., $\mathcal{E} \subseteq \bigcup_{S \in \mathcal{I}} S$, and subject to this constraint, (ii) the number of sets in $\mathcal{I}$ is as small as possible. Set Cover is a well-studied problem with applications in many areas, including operations research [7], information retrieval and data mining [14], web host analysis [2], and many others.

Although the problem is NP-hard, a simple greedy algorithm is guaranteed to report a solution of size at most $O(\ln n)$ larger than the optimum. The algorithm is highly efficient and surprisingly accurate, with the reported solution size often within the 10% of the optimum on typical data sets [7]. However, it has been observed that, due to its sequential nature, the greedy algorithm is significantly less efficient when implemented on hierarchical, parallel and distributed architectures, which are commonly used nowadays for processing massive amounts of data. As a result, there has been considerable work on algorithms for Set Cover that are optimized for external memory [3], streaming [14,6], and cluster computing [2] architectures.

In this paper we consider Set Cover in three related computational models:

1. *Streaming model:* In this model, the sets $S_1, \ldots, S_m$ are stored consecutively in a read-only repository. An algorithm can access the sets by performing sequential scans of the repository. However, the amount of read-write memory available to

the algorithm is limited, and is smaller than the input size (which could be as large as mn). The objective is to design an algorithm that performs few passes over the data, and uses as little memory as possible.

2. *Two-party communication model:* In this model, the sets are partitioned between two parties, Alice and Bob. Without loss of generality we can assume that Alice holds $S_1, \ldots, S_{m/2}$, while Bob holds $S_{m/2+1}, \ldots, S_m$. The parties communicate by exchanging messages, with Alice sending her messages to Bob during the odd rounds, and Bob sending his messages to Alice during the even rounds. The objective is to design a communication protocol to find a minimum cover of $\mathcal{E}$ that terminates in a few rounds such that the total length of the exchanged messages is as small as possible.
3. *Multi-party communication model:* In this model, the sets are partitioned among p parties that are not allowed to communicate with each other. However, there is a *coordinator* that communicates with each of the parties in rounds. In odd rounds, the coordinator performs some computation and broadcasts a single message to all of the parties; in even rounds, each party receives the message, performs some local computation, and sends a message back to the coordinator. Moreover, each party executes the same algorithm. The objective is to design a communication protocol to find a minimum cover of $\mathcal{E}$ that terminates in a few rounds and that the total size of the communication is as small as possible.

The first two models are intimately related. Specifically, any p-pass streaming algorithm that uses s bits of storage yields a $(2p-1)$-round communication protocol exchanging at most $(2p-1)s$ bits (see e.g., [8]). Thus, any efficient streaming algorithm induces a good communication protocol, while any lower bound for the communication complexity provides a lower bound for the amount of storage required by a streaming algorithm. Understanding the amount of communication necessary to solve problems in distributed communication complexity settings has been a subject of extensive research over the last few years, see e.g., [15] for an overview.

The Set Cover problem has attracted a fair amount of research over the last few years. The upper and lower bounds for the problem are depicted in Figure 1. Note that the simple greedy algorithm can be implemented by either storing the whole input (in one pass), or by iteratively updating the set of yet-uncovered elements (in at most n passes).

Our results. Our main result is an $O(4^{1/\delta})$ pass, $O(4^{1/\delta}\rho)$-approximation streaming algorithm with $\tilde{O}(m \cdot n^{\delta})$ space[1] for the Set Cover problem, where ρ denotes the approximation factor of an algorithm that solves Set Cover in off-line model. For example, the greedy algorithm yields $\rho = O(\log n)$, while an exponential algorithm yields $\rho = 1$. In particular, setting $\rho = 1$ and $1/\delta = \frac{1}{2}\log\log n - 1$ implies a $\frac{1}{4}\log n$-approximate communication protocol with complexity $mn^{O(1/\log\log n)}$. This matches the lower bound of Nisan [11] up to a factor of $n^{o(1)}$.

Furthermore, we show $\Omega(mn)$ lower bound for the communication complexity of any *deterministic* protocol approximating two-party Set Cover within a constant factor. Thus, the use of randomness is essential in order to achieve our result.

[1] $\tilde{O}(f(n,m))$ is defined as $O(f(n,m) \cdot \log^k f(n,m))$.

Result	Approximation	Passes/rounds	Space/communication	Type
Greedy	$O(\ln n)$	1	$O(m \cdot n)$	deterministic algorithm
	$O(\ln n)$	n	$O(n)$	deterministic algorithm
[14]	$O(\log n)$	$O(\log n)$	$O(n \log n)$	deterministic algorithm
[6]	$O(\sqrt{n})$	1	$\tilde{O}(n)$	deterministic algorithm
[11]	$\frac{1}{2} \log n$	any	$\Omega(m)$	randomized lower bound
This paper	$O(4^{1/\delta}\rho)$	$O(4^{1/\delta})$	$\tilde{O}(m \cdot n^{\delta})$	randomized algorithm[2]
This paper	$O(1)$	any	$\Omega(m \cdot n)$	deterministic lower bound

Fig. 1. Summary of past work and our results. The algorithmic bounds are stated for the streaming model, while the lower bounds are stated for the two-party communication complexity model. We use ρ to denote the approximation factor of an off-line algorithm solving Set Cover, which is $O(\ln n)$ for the greedy algorithm and 1 for the exponential time algorithm. Furthermore, our result holds for any $\delta = \Omega(1/\log n)$.

We also show in Appendix A that our algorithm implies an $O(4^{1/\delta})$-round $O(4^{1/\delta}\rho)$-approximation communication protocol for multi-party communication model which communicates $\tilde{O}(m \cdot n^{\delta} + p \cdot n)$ bits per round.

Our techniques. Our algorithms exploit *random sampling*. Two variants of sampling are used, depending on the size OPT of the minimum cover. If OPT is large, we use *set sampling*, i.e., we sample $O(\text{OPT})$ random sets and include them in the solution. This ensures that all universe elements contained in $(m \log n)/\text{OPT}$ sets are covered with high probability. Since each of the remaining elements is contained in at most $(m \log n)/\text{OPT}$ sets, the space needed to represent the input is reduced.

On the other hand, if OPT is small, the algorithm performs *element sampling*. Specifically, for a parameter $\alpha > 0$, the algorithm selects $O((\text{OPT} \cdot \log m)/\alpha)$ elements and computes a small cover of those elements. This task can be solved using only $O((m \cdot \text{OPT} \cdot \log m)/\alpha)$ space. We then show that any such solution in fact covers a $1 - \alpha$ fraction of the whole universe. Therefore, it suffices to cover the remaining αn elements, which can be done using less space since the universe size becomes smaller. The aforementioned process can be repeated recursively in order to reduce the space complexity to $O(mn^{\delta})$ for any $\delta > 0$. A variant of the latter approach, element sampling, was previously applied in semi-streaming k-Max Coverage problem [9].

Preliminaries. In this paper we consider the Set Cover problem in the *set streaming* model which is based on the following setup appeared in [14].

Definition 1 (Set Streaming Model). *In* set streaming *model we are given $\mathcal{E}$ in advance and sets in $\mathcal{S}$ are revealed in a stream.*

[2] Our algorithms are randomized. Specifically, we assume that the streaming algorithm has access to a random oracle $r(i)$ such that the bits $r(1), r(2), \ldots$ are i.i.d. symmetric Bernoulli variables. The approximation guarantees offered by our algorithms are required to hold with high probability.

In the *off-line* Set Cover model, the universe of elements $\mathcal{E}$ and the collection of sets $\mathcal{S}$ are given all at once to the algorithm. In this paper, we assume that we are able to approximate off-line Set Cover within a factor ρ of its optimal solution. It is known that under $P \neq NP$, ρ cannot be smaller than $c \cdot \ln n$ where c is a constant [13,1]. At the same time, setting $\rho = 1$ (i.e., assuming an exact algorithm for set cover) provides space/approximation trade-offs without running time considerations. In particular, it establishes the "upper bounds on lower bounds", given that communicational complexity tools for proving lower bounds do not take the running time into account.

A trivial *one* pass streaming algorithm for the Set Cover problem is to read the whole stream and store all sets of $\mathcal{S}$ in memory. This leads to a ρ-approximation algorithm with $O(mn)$ space. We refer to this algorithm as Simple-Set-Cover algorithm which is shown in Figure 2 and will be used later in our algorithms. In Section 3, we

Simple-Set-Cover Algorithm $\langle\langle$Set Cover *Problem. Input:* $\langle \mathcal{E}, \mathcal{S} \rangle\rangle$

Store the projection of all sets in $\mathcal{S}$ over $\mathcal{E}$ in memory
Run the off-line algorithm to find a ρ-approximate cover sol
Return sol

Fig. 2. One pass algorithm for set streaming Set Cover$(\mathcal{E}, \mathcal{S})$ using $O(m \cdot n)$ space.

show that any deterministic *constant* pass constant factor approximation algorithm for set streaming Set Cover requires $\Omega(mn)$ space (see Corollary 1). This implies that the trivial Simple-Set-Cover algorithm is tight. Thus to break the $\Omega(mn)$ space barrier of the constant pass algorithms for set streaming Set Cover, we should consider randomized approaches. In Section 2 we give a randomized constant pass algorithm for the problem that uses $o(mn)$ memory space. Moreover, Nisan proved that any randomized protocol of the Set Cover in two-party communication that achieves an approximation ratio better than $\frac{\log n}{2}$ requires $\Omega(m)$ memory space [11].

2 A Constant Pass Algorithm

In this section, we give a randomized algorithm for set streaming Set Cover that has constant number of passes and consumes $\tilde{O}(m \cdot n^{\delta})$ space where δ is an arbitrary *constant* greater than $4/\log n$. To this end, first in Section 2.1, we describe *set sampling* and *element sampling* approaches followed by a two pass randomized (2ρ)-approximation algorithm that uses $\tilde{O}(m \cdot n^{2/3})$ space to solve the Set Cover problem. Then, in Section 2.2, we extend the techniques further to obtain our main result as follows.

Theorem 1 (Main Theorem). *Suppose that there exists a ρ-approximation algorithm for the* Set Cover *problem in the off-line model. For $\delta = \Omega(1/\log n)$, there exists a randomized $O(4^{1/\delta}\rho)$-approximation algorithm to set streaming* Set Cover *with $O(4^{1/\delta})$ number of passes that consumes $O(m \cdot n^{\delta} \log^2 n \log m)$ bit of memory.*

Note that we can assume that $\delta = \Omega(1/\log n)$ because otherwise the approximation guarantee of Theorem 1 will be $\Omega(\sqrt{n})$ and there exists a single pass $4^{1/\delta}$ approximation algorithm in this case [6].

2.1 Sampling Approaches

In this section, we present two key modules in our algorithm: *element sampling* and *set sampling*.

Element Sampling. Let us assume that we are given k, the size of an optimal solution to Set Cover($\mathcal{E}, \mathcal{S}$). Let $\mathcal{E}_{\text{smp}}$ be a subset of $\mathcal{E}$ of size $O(\rho \cdot \frac{k}{\varepsilon} \log m)$ picked uniformly at random where $\varepsilon < 1$. We claim that a ρ-approximate cover $\mathcal{C}_{\text{smp}}$ of $\mathcal{E}_{\text{smp}}$ is an ε-cover of $\mathcal{E}$ *with high probability* where ε-cover is defined as follows.

Definition 2. *A collection of sets $\mathcal{C}$ is an ε-cover of a set of elements $\mathcal{E}$ if $|\mathcal{E} \cap \bigcup_{S \in \mathcal{C}} S| \geq (1-\varepsilon)|\mathcal{E}|$; in other words, $\mathcal{C}$ covers at least $1-\varepsilon$ fraction of $\mathcal{E}$.*

Since we have assumed that an optimal cover of $\mathcal{E}$ is of size k, there exists a cover of size at most k for $\mathcal{E}_{\text{smp}}$ as well. Let $\mathcal{S}_{\text{smp}} = \{S \cap \mathcal{E}_{\text{smp}} \mid S \in \mathcal{S}\}$ be the collection of the intersections of all sets in $\mathcal{S}$ with $\mathcal{E}_{\text{smp}}$. By calling Simple-Set-Cover($\mathcal{E}_{\text{smp}}, \mathcal{S}$), in one pass we can find a ρ-approximate cover of $\mathcal{E}_{\text{smp}}$, $\mathcal{C}_{\text{smp}}$, using $O(m \cdot |\mathcal{E}_{\text{smp}}|) = O(m\rho \cdot \frac{k}{\varepsilon} \log m)$ bits of memory. We say that $\mathcal{E}_{\text{smp}}$ is a **successful** *element sampling* if $\mathcal{C}_{\text{smp}}$ is an ε-cover of $\mathcal{E}$. The following lemma shows that if $\mathcal{E}_{\text{smp}}$ is picked uniformly at random, then with high probability $\mathcal{E}_{\text{smp}}$ is a successful *element sampling*.

Lemma 1 (Element Sampling Lemma). *Consider an instance of* Set Cover *with $\mathcal{E}$ and $\mathcal{S}$ as inputs. Let us assume that an optimal cover of* Set Cover($\mathcal{E}, \mathcal{S}$) *has size at most k. Let $\mathcal{E}_{\text{smp}}$ be a subset of $\mathcal{E}$ of size $\rho \cdot \frac{ck}{\varepsilon} \log m$ chosen uniformly at random and let $\mathcal{C}_{\text{smp}} \subseteq \mathcal{S}$ be a ρ-approximate cover for $\mathcal{E}_{\text{smp}}$. Then $\mathcal{C}_{\text{smp}}$ is an ε-cover for $\mathcal{E}$ with probability at least $1 - \frac{1}{m^{(c-2)}}$.*

Proof: Since an optimal solution of Set Cover($\mathcal{E}, \mathcal{S}$) has size at most k, an optimal solution of Set Cover($\mathcal{E}_{\text{smp}}, \mathcal{S}_{\text{smp}}$) is also of size at most k. Thus a ρ-approximate cover $\mathcal{C}_{\text{smp}}$ for $\mathcal{E}_{\text{smp}}$ is of size at most $k\rho$.

Let $\mathcal{C}'$ be a subset of $\mathcal{S}$ covering less than $1-\varepsilon$ fraction of $\mathcal{E}$. The probability that $\mathcal{C}'$ covers $\mathcal{E}_{\text{smp}}$ is at most $(1-\varepsilon)^{\rho \frac{ck}{\varepsilon} \log m} < \frac{1}{m}^{ck\rho}$. Thus by union bound, the probability that $\mathcal{C}_{\text{smp}}$ covers $\mathcal{E}_{\text{smp}}$ and $\mathcal{C}_{\text{smp}}$ is an ε-cover of $\mathcal{E}$ is at least

$$1 - \left[\sum_{i=1}^{k\rho} \binom{m}{i}\right] \frac{1}{m^{ck\rho}} \geq 1 - \left[\sum_{i=1}^{k\rho} m^i\right] \frac{1}{m^{ck\rho}} \geq 1 - \frac{m^{k\rho+1}}{m^{ck\rho}}$$
$$\geq 1 - \frac{1}{m^{(c-2)k\rho}} \geq 1 - \frac{1}{m^{c-2}}.$$

Note that the term $\sum_{i=1}^{k\rho} \binom{m}{i}$ counts the number of all covers of size at most $k\rho$ which can be possibly returned as a solution to Set Cover($\mathcal{E}_{\text{smp}}, \mathcal{S}_{\text{smp}}$). □

Let $\mathcal{E}_{\text{rem}}$ be the set of elements remained uncovered after picking $\mathcal{C}_{\text{smp}}$ in the first pass. Lemma 1 showed that with high probability $|\mathcal{E}_{\text{rem}}| \leq \varepsilon n$. In the second pass, we cover the set $\mathcal{E}_{\text{rem}}$ by calling Simple-Set-Cover($\mathcal{E}_{\text{rem}}, \mathcal{S}$) using $O(m \cdot \varepsilon n)$ space. These two steps together give a randomized two-pass (2ρ)-approximation for the problem that uses $O(m \cdot \frac{k\rho}{\varepsilon} \log m + m \cdot \varepsilon n)$ bits of memory which can be optimized by setting $\varepsilon = \sqrt{\frac{k\rho \log m}{n}}$. Thus the total required memory of element sampling is $O(m \cdot \sqrt{\rho k n \log m})$.

Theorem 2. *Let $(\mathcal{E}, \mathcal{S})$ be an instance of set streaming* Set Cover. *Assume that an optimal solution to* Set Cover$(\mathcal{E}, \mathcal{S})$ *has size at most k. Then there exists a two pass randomized (2ρ)-approximation algorithm for the problem that uses $O(m \cdot \sqrt{\rho k n \log m})$ bits of memory.*

However, the required memory space of the described algorithm depends on k and it only performs well for small values of k. In the rest, we remove the dependency on k in the memory space of our algorithm by introducing another sampling module.

Set Sampling. In the *set sampling* module, in a single pass, the algorithm picks a subset of $\mathcal{S}$ uniformly at random. In contrast to *element sampling* technique, *set sampling* works effectively for large k. A *set sampling* $\mathcal{S}_{\text{rnd}}$ of size $c\ell \log n$ is **successful** if $\mathcal{S}_{\text{rnd}}$ covers all elements that appear in at least $\frac{m}{\ell}$ sets of $\mathcal{S}$. The following lemma shows that a subset of $\mathcal{S}$ of size $c\ell \log n$ picked uniformly at random is a successful *set sampling* with high probability.

Lemma 2 (Set Sampling Lemma). *Consider an instance $(\mathcal{E}, \mathcal{S})$ of set streaming* Set Cover. *Let $\mathcal{S}_{\text{rnd}}$ be a collection of sets of size $c\ell \log n$ picked uniformly at random. Then, $\mathcal{S}_{\text{rnd}}$ covers all elements of $\mathcal{E}$ that appear in at least $\frac{m}{\ell}$ sets of $\mathcal{S}$ with probability at least $1 - \frac{1}{n^{c-1}}$.*

Proof: Let e be an element of $\mathcal{E}$ that appears in at least $\frac{m}{\ell}$ sets of $\mathcal{S}$. The probability that e is not covered by $\mathcal{S}_{\text{rnd}}$ is at most $(1 - \frac{1}{\ell})^{c\ell \log n} < e^{-c \ln n / \ln 2} = n^{-c/\ln 2}$. Thus the probability that there exists an element of $\mathcal{E}$ that appears in at least $\frac{m}{\ell}$ sets of $\mathcal{S}$ and is not covered by $\mathcal{S}_{\text{rnd}}$ is at most $n \cdot n^{-c/\ln 2} \leq n^{-c+1}$. □

Two pass algorithm. Now we describe a randomized *two-pass* (2ρ)-approximation algorithm for set streaming Set Cover problem that uses $\tilde{O}(m \cdot n^{2/3})$ space. Let k be a parameter to be determined later and let OPT be the size of an optimal solution of Set Cover$(\mathcal{E}, \mathcal{S})$. Consider the following two cases:

1. OPT $\leq k$. In this case we apply the *element sampling* approach to solve Set Cover$(\mathcal{E}, \mathcal{S})$ using $O(m \cdot \sqrt{\rho n k \log m})$ bits (see Theorem 2).
2. OPT $\geq k$. In this case we apply the *set sampling* module. First, we pick a subset $\mathcal{S}_{\text{rnd}}$ of $\mathcal{S}$ of size $ck\rho$ uniformly at random. By Lemma 2, each element e that is not covered by $\mathcal{S}_{\text{rnd}}$ *with high probability* appears in $\frac{m}{\rho k} \cdot \log n$ sets of $\mathcal{S}$. Thus the required space to solve the problem over uncovered elements off-line is $O(n \cdot \frac{m}{\rho k} \log^2 n)$ bits; the total number of elements in projection of $\mathcal{S}$ over uncovered elements is $O(n \cdot \frac{m}{\rho k} \log n)$ and $O(\log n)$ bits is required for representing each of n elements.

Note that the algorithm does not really need to know OPT. It can run both cases in parallel and at the end, report the best solution of these two. Since each of these subroutines requires two passes, the whole algorithm can be done in two passes. Moreover, the total memory space is $O(m \cdot (\sqrt{n\rho k \log m} + \frac{n}{\rho k} \log^2 n))$ which is minimized by letting $k = \frac{1}{\rho}(\frac{n \log^4 n}{\log m})^{1/3}$. Thus it is a randomized two-pass (2ρ)-approximation algorithm for set streaming Set Cover using $O(m \cdot n^{2/3}(\log m \log^2 n)^{1/3})$ bits of memory.

Lemma 3. *There exists a randomized two-pass (2ρ)-approximation algorithm for set streaming* Set Cover *that uses $\tilde{O}(m \cdot n^{2/3})$ bits of memory.*

2.2 Our Algorithm

In this section we show that we can improve the result of Lemma 3 further in terms of required space by applying the sampling modules recursively. Our main claim is that the Recursive-Sample-Set-Cover algorithm described in Figure 3, achieves the guarantees mentioned in Theorem 1. More precisely, Recursive-Sample-Set-Cover$(\mathcal{E}, \mathcal{S}, n, \delta)$ finds an $O(4^{1/\delta}\rho)$-approximate cover of $\mathcal{E}$ in $O(4^{1/\delta})$ passes using $\tilde{O}(m \cdot n^{\delta})$ bits of memory. We prove Theorem 1 at the end of this section.

In Recursive-Sample-Set-Cover$(\mathcal{E}, \mathcal{S}, n, \delta)$, first we check whether $|\mathcal{E}| \leq n^{\delta}$. If $|\mathcal{E}| \leq n^{\delta}$, we call Simple-Set-Cover$(\mathcal{E}, \mathcal{S})$ to find a cover of $\mathcal{E}$ in one pass using $O(m \cdot n^{\delta})$ bits. Otherwise, similar to the two pass algorithm, we combine *set sampling* and *element sampling* modules. However, here we recurse in *element sampling* module. In Recursive-Sample-Set-Cover$(\mathcal{E}, \mathcal{S}, n, \delta)$ we choose a threshold k to decide whether the size of an optimal cover is large or not. By the proper choice of k and the assumption that all sampling modules are successful, we show that **Case 1** in Recursive-Sample-Set-Cover returns an $O(4^{1/\delta}\rho)$-approximate cover if the size of an optimal cover of $\mathcal{E}$ is larger than or equal to k. Similarly, we show that in the case that the size of an optimal cover of $\mathcal{E}$ is smaller than k, **Case 2** of the algorithm returns an $O(4^{1/\delta}\rho)$-approximate cover. Moreover, in **Case 2** of the algorithm, which corresponds to the *element sampling*, we recursively invoke two instances of Recursive-Sample-Set-Cover on element sets of size at most $\frac{|\mathcal{E}|}{n^{\delta/2}}$. At the end, we return the best solution of these two cases.

Lemma 4. *Let $\mathcal{E}'_{\text{smp}}$ and $\mathcal{E}'_{\text{rem}}$ be subsets of $\mathcal{E}'$ as defined in* Recursive-Sample-Set-Cover$(\mathcal{E}', \mathcal{S}, n, \delta)$. *Then $|\mathcal{E}'_{\text{smp}}| = \frac{|\mathcal{E}'|}{n^{\delta/2}}$ and for large enough c, with high probability, $|\mathcal{E}'_{\text{rem}}| \leq \frac{|\mathcal{E}'|}{cn^{\delta/2}}$.*

Proof: Since k is chosen to be $|\mathcal{E}'|/(c^2\rho \cdot n^{\delta} \cdot \log m)$,

$$|\mathcal{E}'_{\text{smp}}| = c\sqrt{\rho|\mathcal{E}'|k\log m} = \frac{|\mathcal{E}'|}{n^{\delta/2}}.$$

We can rewrite $|\mathcal{E}'_{\text{smp}}|$ as

$$c\sqrt{\rho|\mathcal{E}'|k\log m} = c\rho k \log m / \sqrt{\frac{\rho k \log m}{|\mathcal{E}'|}}.$$

Thus by Lemma 1, a ρ-approximate cover of $\mathcal{E}'_{\text{smp}}$ is a $(\sqrt{\frac{\rho k \log m}{|\mathcal{E}'|}})$-cover of $\mathcal{E}'$ with high probability. Hence, with high probability,

$$|\mathcal{E}'_{\text{rem}}| \leq |\mathcal{E}'| \cdot \sqrt{\frac{\rho k \log m}{|\mathcal{E}'|}} = \sqrt{\rho|\mathcal{E}'|k\log m} = \frac{|\mathcal{E}'|}{cn^{\delta/2}}.$$

□

Next we define the **successful** invocation of Recursive-Sample-Set-Cover.

Definition 3. *An invocation of* Recursive-Sample-Set-Cover$(\mathcal{E}', \mathcal{S}, n, \delta)$ *is* successful *if either $|\mathcal{E}'| \leq n^{\delta}$; or $\mathcal{S}_{\text{rnd}}$ and $\mathcal{E}'_{\text{smp}}$ are respectively successful* element sampling *and* set sampling*, and both* Recursive-Sample-Set-Cover$(\mathcal{E}'_{\text{smp}}, \mathcal{S}, n, \delta)$ *and* Recursive-Sample-Set-Cover$(\mathcal{E}'_{\text{rem}}, \mathcal{S}, n, \delta)$ *are successful.*

Recursive-Sample-Set-Cover $\langle\langle$Set Cover *Problem. Input:* $\langle \mathcal{E}, \mathcal{S}, n, \delta \rangle\rangle$

Let $k = |\mathcal{E}|/(c^2\rho \cdot n^\delta \log m)$
If $|\mathcal{E}| \leq n^\delta$
  $\mathrm{sol} \leftarrow$ Simple-Set-Cover$(\mathcal{E}, \mathcal{S})$ $\langle\langle$*In one pass*$\rangle\rangle$
  Return sol

$\langle\langle$***Case 1****: handling* $\mathrm{OPT}(\mathcal{E}, \mathcal{S}) \geq k$ *via* "set sampling" *module*$\rangle\rangle$
Let $\mathcal{S}_{\mathrm{rnd}}$ be a collection of $ck\rho$ sets of $\mathcal{S}$ picked uniformly at random. $\langle\langle$*In one pass*$\rangle\rangle$
If each element of $\mathcal{E} \setminus \bigcup_{S \in \mathcal{S}_{\mathrm{rnd}}} S$ appears in less than $\frac{m \log n}{k}$ sets of $\mathcal{S}$
  $\mathrm{sol}_{\mathrm{rnd}} \leftarrow$ Simple-Set-Cover$(\mathcal{E} \setminus \bigcup_{S \in \mathcal{S}_{\mathrm{rnd}}} S, \mathcal{S})$ $\langle\langle$*In one pass*$\rangle\rangle$
Else $\langle\langle$*Unsuccessful* set sampling$\rangle\rangle$
  $\mathrm{sol}_{\mathrm{rnd}} \leftarrow$ `Invalid`

$\langle\langle$***Case 2****: handling* $\mathrm{OPT}(\mathcal{E}, \mathcal{S}) < k$ *via* "element sampling" *module*$\rangle\rangle$
Sample a set of elements $\mathcal{E}_{\mathrm{smp}}$ of size $c\sqrt{\rho |\mathcal{E}| k \log m}$ uniformly at random
$\mathrm{sol}_{\mathrm{smp}} \leftarrow$ Recursive-Sample-Set-Cover$(\mathcal{E}_{\mathrm{smp}}, \mathcal{S}, n, \delta)$
Let $\mathcal{E}_{\mathrm{rem}} = \mathcal{E} \setminus \bigcup_{S \in \mathrm{sol}_{\mathrm{smp}}} S$ $\langle\langle$*In one pass*$\rangle\rangle$
If $|\mathcal{E}_{\mathrm{rem}}| \leq |\mathcal{E}|/(cn^{\frac{\delta}{2}})$
  $\mathrm{sol}_{\mathrm{rem}} \leftarrow$ Recursive-Sample-Set-Cover$(\mathcal{E}_{\mathrm{rem}}, \mathcal{S}, n, \delta)$
Else $\langle\langle$*Unsuccessful* element sampling$\rangle\rangle$
  $\mathrm{sol}_{\mathrm{rem}} \leftarrow$ `Invalid`

If any of $\mathrm{sol}_{\mathrm{rnd}}$, $\mathrm{sol}_{\mathrm{smp}}$ or $\mathrm{sol}_{\mathrm{rem}}$ is `Invalid`
  Return `Invalid`
Return the best of $(\mathcal{S}_{\mathrm{rnd}} \cup \mathrm{sol}_{\mathrm{rnd}})$ and $(\mathrm{sol}_{\mathrm{smp}} \cup \mathrm{sol}_{\mathrm{rem}})$

Fig. 3. Recursive-Sample-Set-Cover for the Set Cover problem in set streaming model.

Note that in Recursive-Sample-Set-Cover algorithm we only consider the result of successful invocations. To this end, we discard the run of the algorithm as soon as a sampling module fails.

Consider the *recursion tree* of Recursive-Sample-Set-Cover$(\mathcal{E}, \mathcal{S}, n, \delta)$. Each intermediate node in the tree has two children. Moreover, for each leaf of the tree, the number of elements in its corresponding Recursive-Sample-Set-Cover instance is at most n^δ. Thus, we have the following lemma.

Lemma 5. *The height the recursion tree of* Recursive-Sample-Set-Cover$(\mathcal{E}, \mathcal{S}, n, \delta)$ *is at most* $2/\delta$ *and the number of nodes in the tree is less than* $2 \cdot 4^{1/\delta}$. *Moreover, the number of nodes in the recursion tree is* $O(n)$.

Proof: In the root node of the tree, the element size is n and by lemma 4, the element size decreases by a factor of at least $n^{\delta/2}$ at each level of recursion. Thus, in level i we have at most 2^i instances of Recursive-Sample-Set-Cover with element size at most

$n^{1-i\delta/2}$. Moreover, the element size of the corresponding instances of a leaf is at most n^{δ}. Thus, we can compute the height of the tree, h, as follows:

$$n^{1-\frac{h\delta}{2}} \leq n^{\delta} \Longrightarrow (1-\frac{h\delta}{2}) \leq \delta \Longrightarrow h \leq \frac{2(1-\delta)}{\delta} \leq \frac{2}{\delta}$$

Since the height of the tree is at most $2/\delta$, the total number nodes in the tree is at most $2 \cdot 4^{1/\delta}$. Moreover since $\delta = \Omega(\log n)$, the number of nodes in the tree is $O(n)$. □

The following lemma shows that an invocation of Recursive-Sample-Set-Cover is successful with high probability.

Lemma 6. *Consider an invocation of* Recursive-Sample-Set-Cover$(\mathcal{E}, \mathcal{S}, n, \delta)$. *For sufficiently large c, the invocation is successful with high probability.*

Proof of Lemma 6: Consider any particular node of the recursion tree. By Lemma 2 the probability that *set sampling* performed at that node is successful is at least $1 - \frac{1}{n^{c-1}}$ and by Lemma 1, the probability that the *element sampling* performed at that node is successful is at least $1 - \frac{1}{m^{c-2}}$. Therefore, by union bound over all the nodes in the recursion tree and using the fact that the number of nodes in the recursion tree is $O(n)$ (See Lemma 5), an invocation of the subroutine is successful with probability at least $1 - O(n) \cdot \frac{2}{n^{c-2}} \geq 1 - \frac{1}{O(n^{c-3})}$. □

In the rest we compute the number of passes, approximation guarantee and the required space of Recursive-Sample-Set-Cover$(\mathcal{E}, \mathcal{S}, n, \delta)$.

Lemma 7. *The number of passes in* Recursive-Sample-Set-Cover$(\mathcal{E}, \mathcal{S}, n, \delta)$ *is* $O(4^{1/\delta})$.

Proof: We show that the number of passes the algorithm makes in each node of the recursion tree of Recursive-Sample-Set-Cover is at most 3. Therefore, by Lemma 5 the total number of passes of Recursive-Sample-Set-Cover is $O(4^{1/\delta})$.

In each leaf node which corresponds to an invocation of Recursive-Sample-Set-cover with element size at most n^{δ}, we call Simple-Set-Cover and it is done in one pass. For intermediate nodes, the algorithm has at most the following three passes.

- In the first pass, the algorithm picks $\mathcal{S}_{\mathrm{rnd}}$. In the meantime, it maintains the set of uncovered elements by so far selected sets. Moreover, for each uncovered element e, it stores the number of sets in $\mathcal{S}$ containing e. These numbers are used to decide whether the *set sampling* is successful.
- Next, if $\mathcal{S}_{\mathrm{rnd}}$ is successful, then the algorithm makes another pass to find a cover for the elements that are not covered by $\mathcal{S}_{\mathrm{rnd}}$ via Simple-Set-Cover algorithm.
- Then the algorithm samples a set of elements $\mathcal{E}_{\mathrm{smp}}$ and recursively finds a cover of $\mathcal{E}_{\mathrm{smp}}$. Note that in Recursive-Sample-Set-Cover, we return the indices of the sets in the selected cover. Thus, to decide whether the *element sampling* is successful, the algorithm must make a pass to find the uncovered elements, $\mathcal{E}_{\mathrm{rem}}$. If $|\mathcal{E}_{\mathrm{rem}}| \leq \varepsilon n$, the module is successful and we recursively find a cover for $\mathcal{E}_{\mathrm{rem}}$.

□

Lemma 8. *For sufficiently large c,* Recursive-Sample-Set-Cover$(\mathcal{E}, \mathcal{S}, n, \delta)$ *algorithm returns an $O(4^{1/\delta}\rho)$-approximate solution of* Set Cover$(\mathcal{E}, \mathcal{S})$ *with high probability.*

Proof: By Lemma 6, an invocation of Recursive-Sample-Set-Cover is successful with high probability. In the following we only consider successful invocations of Recursive-Sample-Set-Cover and compute the approximation factor for successful runs.

Consider a successful run of Recursive-Sample-Set-Cover($\mathcal{E}', \mathcal{S}, n, \delta$). If $|\mathcal{E}'| \leq n^\delta$, then the solution returned by the subroutine has size at most $\rho \cdot \text{OPT}$ where OPT is the size of an optimal cover of $\mathcal{E}$.

Otherwise, if an optimum solution for this instance has size at least $|\mathcal{E}'|/(c^2\rho \cdot n^\delta \log m)$ (**Case 1**), the size of the cover constructed by the subroutine is at most $c\rho \cdot (|\mathcal{E}'|/c^2\rho \cdot n^\delta \log m) + \rho \cdot \text{OPT} \leq (c+1)\rho \cdot \text{OPT}$, where the first term denotes the size of $\mathcal{S}_{\text{rnd}}$ and the second term denotes the size of the cover the algorithm picked for the elements that are not covered by $\mathcal{S}_{\text{rnd}}$.

If an optimum cover of the instance has size less than $|\mathcal{E}'|/(c^2\rho \cdot n^\delta \log m)$ (**Case 2**), then the union of the covers returned by Recursive-Sample-Set-Cover($\mathcal{E}'_{\text{smp}}, \mathcal{S}, n, \delta$) and Recursive-Sample-Set-Cover($\mathcal{E}'_{\text{rem}}, \mathcal{S}, n, \delta$) is a cover of $\mathcal{E}$ with small size (for precise value, see Equation 1). By Lemma 4, $|\mathcal{E}'_{\text{smp}}| \leq \frac{|\mathcal{E}'|}{n^{\delta/2}}$ and $|\mathcal{E}'_{\text{rem}}| \leq \frac{|\mathcal{E}'|}{cn^{\delta/2}}$. Since the size of an optimal cover of each of $\mathcal{E}'_{\text{smp}}$ and $\mathcal{E}'_{\text{rem}}$ is less than or equal to the size of an optimal cover of $\mathcal{E}'$, in this case

$$\text{Approx}(|\mathcal{E}'|, n, \delta) \leq 2 \times \text{Approx}(\frac{|\mathcal{E}'|}{n^{\delta/2}}, n, \delta). \tag{1}$$

Thus, we can write the following recursive formula for the approximation guarantee of Recursive-Sample-Set-Cover algorithm.

$$\text{Approx}(|\mathcal{E}'|, n, \delta) \leq \begin{cases} \max\{(c+1)\rho, 2 \times \text{Approx}(|\mathcal{E}'|/n^{\delta/2}, n, \delta)\} & \text{if } |\mathcal{E}'| > n^\delta \\ \rho & \text{if } |\mathcal{E}'| \leq n^\delta \end{cases} \tag{2}$$

By Lemma 5, the height of the recursion tree of our algorithm is $2/\delta$. Hence, a successful run of the algorithm returns an $O(4^{1/\delta}\rho)$-approximate cover. □

Lemma 9. *Consider a successful run of* Recursive-Sample-Set-Cover($\mathcal{E}', \mathcal{S}, n, \delta$). *After picking $\mathcal{S}_{\text{rnd}}$, the required memory space to call* Simple-Set-Cover($\mathcal{E}' \setminus \bigcup_{S \in \mathcal{S}_{\text{rnd}}} S, \mathcal{S}$) *is $O(m \cdot n^\delta \log m \log^2 n)$ bits.*

Proof: As defined in Figure 3, $\mathcal{S}_{\text{rnd}}$ is a collection of sets selected uniformly at random and $|\mathcal{S}_{\text{rnd}}| = ck\rho$ where $k = |\mathcal{E}'|/(c^2\rho \cdot n^\delta \log m)$. In a successful *set sampling*, $\mathcal{S}_{\text{rnd}}$ covers all elements that appear in at least $\frac{m}{k\rho} \cdot \log n$ sets of $\mathcal{S}$. Hence the required space to run Simple-Set-Cover($\mathcal{E}' \setminus \bigcup_{S \in \mathcal{S}_{\text{rnd}}} S, \mathcal{S}$) is $|\mathcal{E}'| \cdot \frac{m}{k\rho} \log n \cdot \log n = c^2 m \cdot n^\delta \log^2 n \log m$. Note that the additional $\log n$ in the memeory space is for representing the elements; $\log n$ bits is required to represent each element. □

Lemma 10. Recursive-Sample-Set-Cover($\mathcal{E}, \mathcal{S}, n, \delta$) *uses $O(m \cdot n^\delta \log m \log^2 n)$ bits of memory to solve set streaming* Set Cover($\mathcal{E}, \mathcal{S}$) *where $n = |\mathcal{E}|$.*

Proof: We prove by induction that the space Recursive-Sample-Set-Cover($\mathcal{E}, \mathcal{S}, n, \delta$) requires is less than $c_1(m \cdot n^\delta + |\mathcal{E}|) \log m \log^2 n$ for a large enough constant c_1.

It is straightforward to see that the induction hypothesis holds for $|\mathcal{E}'| \leq n^{\delta}$. In this case we call Simple-Set-Cover($\mathcal{E}', \mathcal{S}$) that can be executed using $m \cdot n^{\delta}$ bits. Lets assume that the induction hypothesis holds for instances with $|\mathcal{E}| < n'$. In the following we show that the induction hypothesis holds for $|\mathcal{E}'| = n'$ too.

In Recursive-Sample-Set-Cover($\mathcal{E}', \mathcal{S}, n, \delta$), first we perform the *set sampling* module and in this case the required space is bounded by the required space to store $\mathcal{S}_{\text{rnd}}$ which is $|\mathcal{S}_{\text{rnd}}| \cdot \log m$ plus the required space to run Simple-Set-Cover($\mathcal{E} \setminus \bigcup_{S \in \mathcal{S}_{\text{rnd}}} S, \mathcal{S}$) which is $O(m \cdot n^{\delta} \log m \log^2 n)$ (see Lemma 9). We assume that the required space for Simple-Set-Cover($\mathcal{E} \setminus \bigcup_{S \in \mathcal{S}_{\text{rnd}}} S, \mathcal{S}$) is $c_2 \cdot m \cdot n^{\delta} \log m \log^2 n$ (c_2 is computed in the proof of Lemma 9). Thus the total space to run *set sampling*

$$\begin{aligned} ck\rho \log m + c_2 \cdot m \cdot n^{\delta} \log m \log^2 n &\leq n^{1-\delta}/c + c_2 \cdot m \cdot n^{\delta} \log m \log^2 n \\ &\leq c_1 \cdot m \cdot n^{\delta} \log m \log^2 n \end{aligned}$$

which holds for large enough c_1. After executing the *set sampling* module, we only need to keep the constructed cover which requires at most $|\mathcal{E}'| \log m$ bits (the size of the cover is at most $|\mathcal{E}'|$ and for each set in the cover we keep its index).

Then we perform the *element sampling* module. To this end, first we run Recursive-Sample-Set-Cover($\mathcal{E}'_{\text{smp}}, \mathcal{S}, n, \delta$) using $c_1(m \cdot n^{\delta} + |\mathcal{E}'_{\text{smp}}|) \log m \log^2 n$ bits (by induction hypothesis). After constructing a cover for $\mathcal{E}'_{\text{smp}}$, we only keep the cover of $\mathcal{E}'_{\text{smp}}$ which requires at most $|\mathcal{E}'_{\text{smp}}| \cdot \log m$ bits. Next, if $\mathcal{E}'_{\text{smp}}$ is a successful *element sampling*, we cover $\mathcal{E}'_{\text{rem}}$ recursively; otherwise, we return `Invalid`.

Thus the required space of Recursive-Sample-Set-Cover($\mathcal{E}, \mathcal{S}, n, \delta$) is

$$\begin{aligned} &\max\{c_1 \cdot m \cdot n^{\delta} \log m \log^2 n, \;\; |\mathcal{E}'| \log m + c_1 \cdot (m \cdot n^{\delta} + \frac{|\mathcal{E}'|}{n^{\delta/2}}) \log m \log^2 n, \quad (3) \\ &\quad (|\mathcal{E}'| + |\mathcal{E}'|/n^{\delta/2}) \log m + c_1 \cdot (m \cdot n^{\delta} + |\mathcal{E}'|/n^{\delta/2}) \log m \log^2 n\} \\ &\quad = (|\mathcal{E}'| + |\mathcal{E}'|/n^{\delta/2}) \log m + c_1 \cdot (m \cdot n^{\delta} + |\mathcal{E}'|/n^{\delta/2}) \log m \log^2 n \\ &\quad \leq c_1 \cdot (m \cdot n^{\delta} + |\mathcal{E}'|) \log m \log^2 n \quad \text{(for large enough } c_1\text{)} \end{aligned}$$

In Equation 3, the first term denotes the required space while the algorithm is running the *set sampling* module. The second term denotes the required space for the case that the execution of the *set sampling* module is completed and the algorithm is running the first recursive call of the *element sampling* module. The last term is the required memory space while the algorithm is running the second recursive call of the *element sampling* module. Thus induction hypothesis holds for $|\mathcal{E}'| = n'$ and the proof is complete.

□

Theorem 1 follows from Lemma 8, Lemma 7 and Lemma 10.

3 Lower Bounds

In this section, we give some lower bound results for the Set Cover problem in the set streaming model. Specifically, we discuss deterministic protocols and we show that one cannot give a constant pass algorithm with $o(mn)$ memory space that achieves a constant factor approximation for set streaming Set Cover. Our lower bound results follow

from some results in the two-party communication model. In particular we consider the following variant of Set Disjointness problem in two-party communication model.

Definition 4 ((Sparse) Set Disjointness Problem). *In* Set Disjointness(n)*, each of Alice and Bob receives a subset of* $\{1, \ldots, n\}$*,* $\mathcal{S}_A$ *and* $\mathcal{S}_B$*. The goal is to determine whether* $\mathcal{S}_A$ *and* $\mathcal{S}_B$ *are disjoint or not. In* Sparse Set Disjointness(n, k)*, each of two parties receives a subset of size at most* k *of* $\{1, \ldots, n\}$ *and the goal is to determine whether their sets intersect or not.*

Set Disjointness(n) is a well-studied problem in communication complexity and it is known that the best protocol (up to constant) in term of bits of communication is the trivial one in which Alice sends her entire input to Bob. Moreover, using the rank method, it has been shown that any deterministic protocol for Sparse Set Disjointness(n, k) requires $\Omega(m \log(n/k))$ bits of communication.

Nisan [11] proved that any *randomized* protocol approximating Set Cover in two-party communication with a factor better than $\frac{\log n}{2}$ has communication complexity $\Omega(m)$. In this section, exploiting the techniques of [11], we get $\Omega(mn)$ lower bound for the memory space of deterministic two-party protocols approximating Set Cover within a constant factor.

Definition 5 (r-covering property [10,11]). *Let* $\mathcal{S}$ *be a collection of subsets of* $\{1, \ldots, n\}$*. The collection* $\mathcal{S}$ *has the* r*-covering property if for every collection* $\mathcal{A} \subseteq \{S \mid S \in \mathcal{S} \text{ or } \overline{S} \in \mathcal{S}\}$ *of size at most* r*,* $\mathcal{A}$ *does not cover* $\{1, \ldots, n\}$ *unless a set* S *and its complement are both selected in* $\mathcal{A}$*.*

Lemma 11 (From [11]). *For any* $r \leq \log n - O(\log \log n)$*, there exists a collection* $\mathcal{S}$ *of subsets of* $\{1, \ldots, n\}$ *that satisfies the* r*-covering property such that* $|\mathcal{S}| \geq e^{n/(r2^r)}$*.*

Combining the known lower bound of Sparse Set Disjointness(n, k) with the r-covering property, we achieve the following lower bound result for deterministic protocols of Set Cover in two-party communication.

Theorem 3. *Any deterministic* α*-approximation protocol for* Set Cover$(\mathcal{E}, \mathcal{S}_A, \mathcal{S}_B)$ *in two-party communication requires* $\Omega(|\mathcal{S}_A \cup \mathcal{S}_B| \cdot |\mathcal{E}|)$ *communication if* $\alpha = O(1)$*.*

Proof: Given an instance (x_A, x_B) of Sparse Set Disjointness(n, k), we construct the following corresponding instance of two-party Set Cover$(\mathcal{E}, \mathcal{S}_A, \mathcal{S}_B)$.
Let $r = 2\alpha$ and let $\mathcal{S} = \{S_1, \ldots, S_n\}$ be a collection of subsets of $\{1, \ldots, p\}$ satisfying r-covering property. By Lemma 11, it is enough to have $|\mathcal{S}| = e^{p/(r2^r)}$ which implies that $p = r2^r \ln n$. Since $r = O(1)$, we have $p = O(\log n)$.

Define $\mathcal{E} = \{1, \ldots, p\}$. Let $\mathcal{S}_A$ be the collection of sets that Alice owns and let $\mathcal{S}_B$ denote the collection of sets owned by Bob. We define $\mathcal{S}_A = \{S_i \mid x_A[i] = 1\}$ and $\mathcal{S}_B = \{\overline{S_i} \mid x_B[i] = 1\}$.
The r-covering property of $\mathcal{S}$ guarantees that the size of an optimal cover of $\mathcal{E}$, $\mathcal{C} \subseteq \mathcal{S}$, is either 2 (the case that $\mathcal{C}$ contains both S and $\overline{S}$ for a set $S \in \mathcal{S}$) or at least r. Note that x_A and x_B intersect iff the size of an optimal cover of $\mathcal{E}$ is 2. Thus any protocol for two party Set Cover$(\mathcal{E}, \mathcal{S}_A, \mathcal{S}_B)$ with approximation ratio smaller than $r/2$ solves Sparse Set Disjointness(n, k) exactly.

It has been shown that Sparse Set Disjointness(n, k) has communication complexity $\Omega(k \log(2n/k))$. If we pick k such that $k = O(n^{1-\epsilon})$ for some constant ϵ, then $|\mathcal{E}| = p = O(\log n) = O(\log \frac{2n}{k})$. Thus by the known lower bound of Sparse Set Disjointness(n, k) in two party communication, two-party Set Cover$(\mathcal{E}, \mathcal{S}_A, \mathcal{S}_B)$ requires $\Omega(k \log(2n/k)) = \Omega(k \cdot p) = \Omega((|\mathcal{S}_A| + |\mathcal{S}_B|) \cdot |\mathcal{E}|)$ bits of communication. □

Corollary 1. *Any deterministic constant factor approximation algorithm for set streaming* Set Cover *with constant number of passes requires $\Omega(mn)$ space.*

The following is based on the lower bound of [11].

Corollary 2. *Any randomized constant pass algorithm that approximates set streaming* Set Cover$(\mathcal{E}, \mathcal{S}_A, \mathcal{S}_B)$ *within a factor smaller than $\frac{\log n}{2}$ uses $\Omega(|\mathcal{S}_A \cup \mathcal{S}_B|)$ space.*

Acknowledgment. The work was supported in part by NSF grants CCF-1161626 and CCF-1065125, DARPA/AFOSR grant FA9550-12-1-0423, Packard Foundation, Simons Foundation and MADALGO — Center for Massive Data Algorithmics — a Center of the Danish National Research Foundation.

References

1. Noga Alon, Dana Moshkovitz, and Shmuel Safra. Algorithmic construction of sets for k-restrictions. *ACM Transactions on Algorithms*, 2(2):153–177, 2006.
2. Flavio Chierichetti, Ravi Kumar, and Andrew Tomkins. Max-cover in map-reduce. In *Proc. of WWW*, pages 231–240. ACM, 2010.
3. Graham Cormode, Howard Karloff, and Anthony Wirth. Set cover algorithms for very large datasets. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 479–488. ACM, 2010.
4. Danny Dolev and Tomás Feder. Multiparty communication complexity. In *Proc. of IEEE FOCS*, pages 428–433. IEEE, 1989.
5. Pavol Ďuriš and José DP Rolim. Lower bounds on the multiparty communication complexity. *Journal of Computer and System Sciences*, 56(1):90–95, 1998.
6. Yuval Emek and Adi Rosén. Semi-streaming set cover. In *Proc. of ICALP*, 2014.
7. Tal Grossman and Avishai Wool. Computational experience with approximation algorithms for the set covering problem. *European Journal of Operational Research*, 101(1):81–92, 1997.
8. Sudipto Guha and Andrew McGregor. Tight lower bounds for multi-pass stream computation via pass elimination. In *Proc. of ICALP*, pages 760–772. Springer, 2008.
9. Ravi Kumar, Benjamin Moseley, Sergei Vassilvitskii, and Andrea Vattani. Fast greedy algorithms in mapreduce and streaming. In *Proc. of SPAA*, pages 1–10. ACM, 2013.
10. Carsten Lund and Mihalis Yannakakis. On the hardness of approximating minimization problems. *Journal of the ACM*, 41(5):960–981, 1994.
11. Noam Nisan. The communication complexity of approximate set packing and covering. In *Proc. of ICALP*, pages 868–875. Springer, 2002.
12. Jeff M Phillips, Elad Verbin, and Qin Zhang. Lower bounds for number-in-hand multiparty communication complexity, made easy. In *Proc. of ACM-SIAM SODA*, pages 486–501. SIAM, 2012.

13. Ran Raz and Shmuel Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability pcp characterization of np. In *Proc. of ACM STOC*, pages 475–484. ACM, 1997.
14. Barna Saha and Lise Getoor. On maximum coverage in the streaming model & application to multi-topic blog-watch. In *SDM*, pages 697–708, 2009.
15. David P Woodruff and Qin Zhang. When distributed computation is communication expensive. In *Distributed Computing*, pages 16–30. Springer, 2013.

A Multiparty Communication Algorithm for Set Cover

In this section, we describe how our algorithm can be applied in the *multi-party communication* model where the input is distributed among a set of p parties and the goal is to compute a function over the input, while minimizing the total amount of communication. It is assumed that there is *coordinator* which can communicate with each of the parties, however the parties do not communicate with each other directly. This model has been widely studied before (see for example [5,4,12]). Note that this model is an example of the *number-in-hand* model in which each party sees its own input, as opposed to the *number-on-forehead* model where each party can see the inputs of all the other parties except his own.

Our model. In *coordinator* model there are p parties $P_1, \ldots, P_p$ and one coordinator. The coordinator can communicate with each of the parties, but the parties cannot communicate with each other. Also, only synchronous executions are considered: in even rounds, each party receives a message from the coordinator, performs some local computation, and sends a message back to the coordinator. In odd rounds, the coordinator receives messages from each party, performs some computation and broadcasts the same message to all of the parties. We consider a restricted variant of the coordinator model in which each party executes the same algorithm.

The problem and our result. Let $\mathcal{E} = \{1, \cdots, n\}$ be the element set and let $\mathcal{S}$ be a collection of m sets and let $\mathcal{S}_1, \ldots, \mathcal{S}_p$ be a partitioning of $\mathcal{S}$ such that the party i only has the collection $\mathcal{S}_i$. The goal of the algorithm is for the coordinator to output the indices of the sets in $\mathcal{S}$ that constitute a minimum cover for $\mathcal{E}$.

We are interested in the total number of rounds, approximation factor and total amount of communication per round. Note that the communication of odd rounds is counted as the size of the single message broadcasted from the coordinator, and the communication of even rounds is counted as the total size of the messages from all the parties to the coordinator. Assuming the described model, we have the following theorem which mainly follows from Theorem 1.

Theorem 4. *There is a randomized $O(4^{1/\delta})$-round, $O(4^{1/\delta}\rho)$-approximation algorithm to* Set Cover$(\mathcal{E}, \mathcal{S})$ *with total communication of $\tilde{O}(m \cdot n^{\delta} + p \cdot n)$ in each round.*

Here ρ is the approximation ratio of the off-line Set Cover achieved by the coordinator.

Our algorithm. We can show that it is possible for the coordinator to run the Recursive-Sample-Set-Cover such that its total communication with the parties is $\tilde{O}(m \cdot n^{\delta})$ in each

round. The algorithm only needs to access the sets in one of the following three cases.
If $|\mathcal{E}| \leq n^{\delta}$:

1. First the coordinator broadcasts $\mathcal{E}$ to the parties using $O(n^{\delta} \log n)$ bits.
2. Each party P_i, sends the projection of it sets on $\mathcal{E}$ back to the coordinator, i.e., $\{S \cap \mathcal{E} \mid S \in \mathcal{S}_i\}$. This needs at most $O(mn^{\delta} \log n)$ bits of communication.

Set Sampling:

1. To choose a collection of $ck\rho$ sets uniformly at random, $\mathcal{S}_{\text{rnd}}$, the coordinator sends a constant size message to all parties to initiate the set sampling module.
2. Then each party P_i generates and sends a random number corresponding to each of its sets which is a vector of size $|\mathcal{S}_i|$ of random numbers. Note that it is enough for the random numbers to be in the range $(1, \ldots, m^c)$ for some constant c so that with high probability, the numbers for all the sets in $\mathcal{S}$ do not collide. This needs at most $O(m \log m)$ bits of communication.
3. The coordinator finds a threshold thr such that there are exactly $ck\rho$ numbers below thr among the received numbers and broadcasts thr. It requires $O(\log m)$ bits.
4. Each party P_i sends back a bit vector of size n showing which elements are covered by the sets in $\mathcal{S}_i$ whose assigned random number is less than thr. This needs at most $O(p \cdot n)$ bits of communication.
5. The coordinator broadcasts the set of uncovered elements $\mathcal{E}_{\text{rem}}$ using $O(n \log n)$ bits.
6. Each party P_i returns for each uncovered element e, the number of sets in $\mathcal{S}_i$ that contains e. This needs at most $O(p \cdot n \log m)$ bits of communication.
7. The coordinator checks whether $\mathcal{S}_{\text{rnd}}$ is successful. In the case of success, it sends "success" message using $O(1)$ bits.
8. In the case of success, each of the parties projects its sets on the uncovered elements and sends it back to the coordinator. Then coordinator solves the Set Cover problem over the uncovered elements off-line. By Lemma 9, this needs at most $O(mn^{\delta} \log m \log^2 n)$ bits of communication.

Element Sampling. The element sampling requires recursively invoking the algorithm for the instances of the Set Cover problem with smaller element size. The coordinator samples a set of elements $\mathcal{E}_{\text{smp}}$ and solves the problem recursively for $\mathcal{E}_{\text{smp}}$. Then the coordinator checks whether $\mathcal{E}_{\text{smp}}$ was successful and in this case of success, it solves the problem recursively for the set of uncovered elements $\mathcal{E}_{\text{rem}}$.

By the analysis of Recursive-Sample-Set-Cover in Section 2, it is straightforward to check that the total communication in each round is $\tilde{O}(m \cdot n^{\delta} + p \cdot n)$. Also the total number of rounds is constant per each recursive call of Recursive-Sample-Set-Cover. Therefore similar to the proof of Lemma 7, the total number of rounds of the algorithm is $O(4^{1/\delta})$. Hence the total communication of the algorithm is $\tilde{O}(4^{1/\delta}(m \cdot n^{\delta} + p \cdot n))$.