Almost Optimal Streaming Algorithms for Coverage Problems

MohammadHossein Bateni
Google Research

Hossein Esfandiari
University of Maryland

Vahab Mirrokni
Google Research

ABSTRACT

Maximum coverage and minimum set cover problems—here collectively called coverage problems—have been studied extensively in streaming models. However, previous research not only achieve suboptimal approximation factors and space complexities, but also study a restricted set-arrival model which makes an explicit or implicit assumption on oracle access to the sets, ignoring the complexity of reading and storing the whole set at once. In this paper, we address the above shortcomings, and present algorithms with improved approximation factor and improved space complexity, and prove that our results are almost tight. Moreover, unlike most of previous work, our results hold in a more general edge-arrival model.

More specifically, consider an instance with n sets, together covering m elements. Information arrives in the form of "edges" from sets to elements (denoting membership) in arbitrary order.

1. We present (almost) optimal approximation algorithms for maximum coverage and minimum set cover problems in the streaming model with an (almost) optimal space complexity of $\tilde{O}(n)$; i.e., the space is *independent of the size of the sets or the size of the ground set of elements*. These results not only improve the best known algorithms for the set-arrival model, but also are the first such algorithms for the more powerful *edge-arrival* model.

2. In order to achieve the above results, we introduce a new general sketching technique for coverage functions: One can apply this sketching scheme to convert an α-approximation algorithm for a coverage problem to a $(1-\varepsilon)\alpha$-approximation algorithm for the same problem in streaming model.

3. We show the significance of our sketching technique by ruling out the possibility of solving coverage problems via accessing (as a black box) a $(1 \pm \varepsilon)$-approximate oracle (e.g., a sketch function) that estimates the coverage function on any subfamily of the sets. Finally, we show that our streaming algorithms achieve an almost optimal space complexity.

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

1. INTRODUCTION

Maximum coverage and minimum set cover problems—here collectively called *coverage problems*—are among the most fundamental problems in optimization and computer science. Coverage problems have a variety of machine-learning and data-mining applications (for examples in data summarization and web mining, see [14, 1, 12, 38, 9]). Solving such problems has become increasingly important for various real-world large-scale data-mining applications where due to the sheer amount of data, either the computation has to be done in a distributed manner [14, 17, 11, 35, 38, 27, 37], or the data is presented and needs to be analyzed in a stream [9, 6, 44, 41, 19, 13].

These problems have been explored extensively in the literature, but despite development of several scalable algorithms, the existing approaches still suffer from a few shortcomings. First of all, most previously studied models make an explicit or implicit assumption on having oracle access to each set in its entirety. This assumption, in particular, ignores the computational complexity of reading the whole set, or computing the marginal impact of adding a subset to the solution (i.e., computing union and intersection of family of subsets). For instance, in the streaming setting, this assumption is implied in the extensively studied set-arrival model [44, 19, 18, 13]. Such models are less realistic since all the information of each set need to be gathered together. The set-arrival setting directly translates to the vertex-arrival setting in graph streaming[1], which is less interesting than the popular edge-arrival setting [4, 5, 7, 15, 21, 30, 31]. Secondly, current streaming algorithms often achieve suboptimal approximation guarantees compared to the offline optimum or do not have the best space complexities in terms of the number of sets in the input.[2]

In this paper, we aim to address the above issues. We develop streaming algorithms that achieve optimal approximation guarantees as well as optimal space complexities for coverage problems without any oracle-access assumptions. Moreover, our algorithm works in the (more general) edge-arrival streaming model. At the core of our analysis lies a simple, yet subtle sketching technique. In order to demonstrate the power of this technique, we show why natural sketching approaches do not work well. We also demonstrate that oracle access to a noisy estimator for the coverage function is not sufficient. We first present more formal definitions before elaborating on these results.

1.1 Preliminaries

Coverage Problems.

We study three related *coverage* problems. The setting includes a ground set $\mathcal{E}$ of m elements, and a family $\mathcal{S} \subseteq 2^{\mathcal{E}}$ of n subsets of the elements (i.e., $n = |\mathcal{S}|$ and $m = |\mathcal{E}|$).[3] *The coverage function* $\mathcal{C}$ is defined as $\mathcal{C}(S) = | \cup_{U \in S} U|$ for any subfamily $S \subseteq \mathcal{S}$ of subsets. In the *k-cover* problem, given a parameter k, the goal is to find k sets in $\mathcal{S}$ with the largest union size. We sometimes use Opt_k to denote the size of the union for the optimum solution. In the *set cover* problem, the goal is to pick the minimum number of sets from $\mathcal{S}$ such that all elements in $\mathcal{E}$ are covered. We also study a third problem: In the *set cover with* λ *outliers* problem[4], the goal is to find the minimum number of sets covering at least a $1 - \lambda$ fraction of the elements in $\mathcal{E}$.

Coverage problems may be modeled as a bipartite graph G, where $\mathcal{S}$ corresponds to one part of the vertices, and $\mathcal{E}$ corresponds to the other part. A vertex representing the set $S \in \mathcal{S}$ has $|S|$ edges in G, one to each element $i \in S$. For simplicity, we assume that there is no isolated vertex in $\mathcal{E}$. For a subset S of vertices in a graph G, let $\Gamma(G, S)$ denote the set of neighbors of S. When G is the graph corresponding to the original coverage instance, we have $\mathcal{C}(S) = |\Gamma(G, S)|$ if S is a subfamily of the sets $\mathcal{S}$.

In the offline setting, a simple greedy algorithm achieves $1 - \frac{1}{e}$ approximation for k-cover and $\log m$ approximation algorithm for the set cover problem.[5] Moreover, improving these approximation factors are impossible unless NP has slightly superpolynomial time algorithm [22].

Streaming models.

In the *streaming* model, we focus on the so-called *edge-arrival* model as opposed to the more studied *set-arrival* (aka *vertex-arrival*) model. In the former, edges arrive one by one, so we get to know about the set-element membership relations one at a time, whereas in the latter, sets arrive and bring with them a list of their elements. The number of passes allowed for processing the data is crucial and may change the nature of the problem.

The $(1 \pm \varepsilon)$*-approximate oracle..*

We say $\mathcal{C}_\varepsilon$ is a $(1 \pm \varepsilon)$*-approximate oracle* to coverage function $\mathcal{C}$ if, given a subfamily of sets, it gives us an estimate of their union size within $1 \pm \varepsilon$ precision. In other words, $\mathcal{C}_\varepsilon$ estimates the coverage function $\mathcal{C}$ on any subfamily of the sets as a black box; i.e., for any subset $S \subseteq \mathcal{S}$, we have

$$(1 - \epsilon)\mathcal{C}_\epsilon(S) \leq \mathcal{C}(S) \leq (1 + \epsilon)\mathcal{C}_\epsilon(S).$$

1.2 Related work

Coverage problems have been studied extensively in the context of set-arrival models [6, 44, 41, 19, 13]. Most of these give suboptimal approximation guarantees. In particular, Saha and Getoor [44] provide a $\frac{1}{4}$-approximation algorithm for k-cover in one pass using $\tilde{O}(m)$ space. The same technique gives a $\Theta(\log m)$ approximation algorithm for set cover in $\Theta(\log m)$ passes, using $\tilde{O}(m)$ space. On the hardness side, interestingly, Assadi et al. [6] show that there is no α-approximation one-pass streaming algorithm for set cover using $o(nm/\alpha)$ space. Demaine et al. [18] provide (for any positive integer r) a $4^r \log m$-approximation algorithm for the set cover problem in 4^r passes using $\tilde{O}(nm^{1/r} + m)$

[1]Modeled as a bipartite graph where vertices on one side corresponds to the sets and vertices on the other side corresponds to elements. See Preliminaries for a formal definition.

[2]We focus on the regime where the number of the element (i.e., the size of the ground set) is significantly larger than the number of sets, hence the importance of having bounds in terms of the number of sets rather than elements.

[3]There are two separate series of work in this area. We use the convension of the submodular/welfare maximization formulation [8], whereas the hypergraph-based formulation [44] typically uses n, m in the opposite way.

[4]This is sometimes called the $(1 - \lambda)$-partial cover problem in the literature.

[5]Unless otherwise specified, we use the wide-spread convension for approximation ratios: factors larger than one for minimization problems and factors smaller than one for maximization problems.

space[6]. Recently, Har-Peled et al. improves this result and provide a p-pass $O(p \log m)$-approximation algorithm in $\tilde{O}(nm^{O(1/p)} + m)$ space[6]. Indeed, all the above results hold only for the set-arrival model.

Often in the graph streaming problems, while the size of the input is $\tilde{O}(|E|)$ for a graph $G(V, E)$, the solution size may be as large as $\Omega(|V|)$. The best hope then is to find the solution in $\tilde{O}(|V|)$ space. Algorithms fitting this description are called *semi-streaming* [39], and many graph problems have been studied in this setting [2, 3, 20, 23, 24, 32, 33, 34]. On the other hand, the extensive work on edge-arrival streaming [4, 5, 7, 15, 21, 30, 31] had not (prior to our owrk) studied coverage problems.

1.3 Results and techniques

As our main result, we address the aforementioned shortcomings of existing algorithms for coverage problems. These results are summarized in Table 1. This paper is *the first to study the problem in the edge-arrival model*, and present tight results for these problems.

1.3.1 Streaming results

We present almost tight streaming algorithms for coverage problems. The following theorem states our main results formally.

Theorem 1.1. *In the edge-arrival streaming model, for any arbitrary $\varepsilon \in (0, 1]$, there exist*

- *(See Thm 3.1) a single-pass $(1 - \frac{1}{e} - \varepsilon)$-approximation algorithm for k-cover using $\tilde{O}(n)$ space;*
- *(See Thm 3.3) a single-pass $(1+\varepsilon) \log \frac{1}{\lambda}$-approximation algorithm for set cover with λ outliers using $\tilde{O_\lambda}(n)$ space; and*
- *(See Thm 3.4) a p-pass $(1 + \varepsilon) \log m$-approximation algorithm for set cover using $\tilde{O}(nm^{O(\frac{1}{p})} + m)$ space.*

The above are the first such results for coverage problems in the streaming edge-arrival model. Moreover, they improve the approximation factor of previously known results for the set-arrival model [44, 41, 19, 13]. (However, in certain cases, the space complexities may be incomparable, say, $\tilde{O}(n)$ versus $\tilde{O}(m)$.[7]) In fact, our result for streaming set cover gives an exponential improvement over Demaine et al. [18] on both approximation factor and number of rounds given the same space. See Table 1 for comparison to previous work. Recently, Har-Peled et al. (Theorem 2.6 in [25]) provide a p-pass $O(p \log m)$-approximation algorithm in $\tilde{O}(nm^{O(1/p)}+m)$ space in the set-arrival model. Notice that our results for streaming set cover provide a better approximation factor—i.e., $(1+\varepsilon) \log m$ versus $O(p \log m)$—in the same space and number of passes, while handling the more general edge-arrival model.

On the hardness side, we show that any $\frac{1}{2} + \varepsilon$-approximation streaming algorithm for k-cover requires $\Omega(n)$ space. This holds even for streaming algorithms with several passes.

[6]The space bounds claimed in [18, 25] assume $m = O(n)$, hence stated differently.

[7]Indeed, either m or n may be larger in practice [16]. See also Footnotes 2 and 6.

Theorem 1.2. *Any $\frac{1}{2} + \varepsilon$-approximation multi-pass streaming algorithm for k-cover requires $\Omega(n)$ space in total.*

In a simultaneous and independent work, McGregor and Vu [36] present a single-pass $1 - 1/e - \epsilon$ approximation algorithm for the k-cover problem in the streaming setting with $\tilde{O}(n)$ space, using a different approach: They directly analyze the behavior of the greedy algorithm on a specific noisy sketch, while we provide a sketch that translates any α-approximation algorithm for k-cover to an $(\alpha - \epsilon)$-approximation streaming algorithm using $\tilde{O}(n)$ space.

1.3.2 Sketching technique

The main technique at the heart of our results is a powerful sketching to summarize coverage functions. As its main property, we show that any α-approximate solution to k-cover on this sketch is an $(\alpha - \varepsilon)$-approximate solution to k-cover on the original input with high probability; see Theorem 2.7. Interestingly, this sketch requires only $\tilde{O}(n)$ space. Our sketch is fairly similar to ℓ_0 sketches [16], which are essentially defined to estimate the value of coverage functions; see Appendix C for a formal definition. Indeed, one may maintain n instances of the ℓ_0 sketch, and estimate the value of the coverage function of a single feasible solution of size k with high probability. However, having $\binom{n}{k}$ different choices for a solution of size k leads to a huge blow-up on the failure probability of at least one such solution. In Appendix C, we show a straightforward analysis to approximate k-cover using ℓ_0 sketches with $\tilde{O}(nk)$ space, which is quite larger than our sketch.

All the algorithms presented here construct $\tilde{O}(1)$ independent instances of the sketch and then solve the problem without any other direct access to the input. The simplicity of our sketch enables its efficient construction and fast implementation of the resulting algorithms. Interestingly, this technique provides almost tight approximation guarantees. We remark that all the algorithms presented in this work have success probabilities $1 - \frac{1}{n}$; i.e., they may fail to produce the claimed solution with probability $\frac{1}{n}$. For simplicity we do not repeat this condition elsewhere.

Finally, in an accompanied paper, we also show how to apply this to distributed models, and design scalable distributed algorithms for covering problems. There we also confirm the effectiveness of this algorithm empirically on real data sets [10].[8]

1.3.3 A $(1 \pm \varepsilon)$-approximate oracle is not sufficient

There are several sampling or sketching techniques that can be used to develop a $(1 \pm \varepsilon)$-approximate oracle $\mathcal{C}_\varepsilon$ to the coverage function. One might hope that a black-box access to such an oracle could be used as a subroutine in developing approximation algorithms with good approximation guarantees. Here, we show that this is not possible.

Theorem 1.3. *Any α-approximation algorithm for k-cover via oracle $\mathcal{C}_\varepsilon$ requires $\exp\left(\Omega(n\varepsilon^2\alpha^2 - \log n)\right)$ queries to the oracle.*

In particular, for any constant $\varepsilon > 0$, there is no polynomial-time $n^{-0.49}$ approximation algorithm for k-cover given a $(1 \pm \varepsilon)$-approximate oracle $\mathcal{C}_\varepsilon$. This improves upon a similar hardness result for submodular functions [26]—and

[8]We decided to remove this part of the paper due to space constraints, and focus on the streaming applications.

Problem	Credit	# passes	Approximation	Space	Arrival
k-cover	[44]	1	$1/4$	$\tilde{O}(m)$	set
k-cover	[9]	1	$1/2$	$\tilde{O}(n+m)$	set
k-cover	Here	1	$1-1/e-\varepsilon$	$\tilde{O}(n)$	edge
Set cover w. outliers	[19, 13]	p	$O(\min(n^{\frac{1}{p+1}}, e^{-\frac{1}{p}}))$	$\tilde{O}(m)$	set
Set cover w. outliers	Here	1	$(1+\varepsilon)\log\frac{1}{\lambda}$	$\tilde{O}_\lambda(n)$	edge
Set cover	[13, 44]	p	$(p+1)m^{\frac{1}{p+1}}$	$\tilde{O}(m)$	set
Set cover	[18]	4^r	$4^r\log m$	$\tilde{O}(nm^{\frac{1}{r}}+m)$	set
Set cover	[25]	p	$O(p\log m)$	$\tilde{O}(nm^{O(\frac{1}{p})}+m)$	set
Set cover	Here	p	$(1+\varepsilon)\log m$	$\tilde{O}(nm^{O(\frac{1}{p})}+m)$	edge

Table 1: Comparison of results in streaming models. Note that all our results for edge arrival model also hold for the set arrival model.

not for coverage functions. Our proof technique here might be of independent interest. (See details in Appendix 5.)

In order to prove Theorem 1.3, first we define a problem called *k-purification* for which we show that any randomized algorithm requires $\delta \exp\left(\Omega(\frac{\varepsilon^2 k^2}{n})\right)$ oracle queries to succeed with probability δ. In a k-purification problem instance, we are given a random permutation of n items, with k gold and $n-k$ brass items. The types of individual items are not known to us. We merely have access to an oracle $\mathsf{Pure}_\varepsilon(S)$ for $S \subseteq [1,n]$ defined as

$$\begin{cases} 0 & \text{if } \frac{k|S|}{n} - \varepsilon\left(\frac{k|S|}{n} + \frac{k^2}{n}\right) \le \mathsf{Gold}(S) \le \frac{k|S|}{n} + \varepsilon\left(\frac{k|S|}{n} + \frac{k^2}{n}\right), \\ 1 & \text{otherwise,} \end{cases}$$

where $\mathsf{Gold}(S)$ is the number of gold items in S. The goal in this problem is to find a set S such that $\mathsf{Pure}_\varepsilon(S) = 1$. The hardness proof is then based on a reduction between k-purification and k-cover.

1.4 Organization

We next present the core idea behind our sketching technique and then explain our algorithms in Section 3. Due to space constraints, most proofs and discussions appear in the appendix. In particular, we present in the appendix our negative result for the black-box usage of $(1 \pm \varepsilon)$-approximate oracles.

2. SKETCHING FOR COVERAGE PROBLEMS

In this section we present a sketch $H_{\le n}$ to approximate k-cover. Specifically, we show that any α-approximate solution to k-cover on $H_{\le n}$ is an $\alpha - O(\varepsilon)$-approximate solution on the input graph, with high probability (see Theorem 2.7). Crucially $H_{\le n}$ uses only $\tilde{O}(n)$ space. In order to define and prove the properties of $H_{\le n}$, we introduce two intermediary sketches H_p and H'_p, where $p \in [0,1]$ is a parameter to be fixed later on.

In this section, we define the sketch in mathematical terms and establish its desirable properties. Then in the following section, we discuss the intricacies of building and using it in the streaming model.

Let h be a hash function mapping elements $\mathcal{E}$ to real numbers in $[0,1]$. First we throw away from the bipartite graph G any element whose hash value exceeds p. This constructs H_p. In Lemma 2.3 we show that, for sufficiently large p, any α-approximate solution to k-cover on H_p is an $\alpha - O(\varepsilon)$-approximate solution on G, with high probability. Unfortunately, the number of edges in H_p may be $\Omega(nk)$.

Next we enforce an upper bound (defined below in terms of n, k, ε) on the degree of elements in H_p, by arbitrarily removing edges as necessary. This constructs H'_p. Again for a sufficiently large choice of p, any α-approximate solution to k-cover on H'_p is an $\alpha - O(\varepsilon)$-approximate solution on G, with high probability. Interestingly, if we select p wisely, H'_p requires only $\tilde{O}(n)$ space. However, this p depends on the value of the optimum solution and may not be accessible to the algorithm while constructing the sketch. To resolve this issue, we define $H_{\le n}$ with a similar structure as H'_p, such that it always has $\tilde{O}(n)$ edges (see Definition 2.1). We remark that this conceptual description can be turned into efficient implementations in several computational frameworks. Next comes the formal definitions of our sketch.

Let us overload the notation $h(.)$ such that $h(e)$, for an edge e, denotes the value of h on the endpoint of e in $\mathcal{E}$. For a fixed parameter p, we define H_p to be the subgraph of G induced by all vertices in $\mathcal{S}$ and the vertices in $\mathcal{E}$ with h less than p. In other words, H_p contains an edge e if and only if $h(e) \le p$. Let H'_p be a maximal subgraph of H_p such that the degree of the vertices of H'_p in part $\mathcal{E}$ is at most $\frac{n\log(1/\varepsilon)}{\varepsilon k}$; as necessary we throw away edges arbitrarily. Below we define $H_{\le n}(k, \varepsilon, \delta'')$ based on H'_p. The former is the sketch used in all our algorithms.

Definition 2.1. *For simplicity of notation, we set* $\delta = \delta'' \log\log_{1-\varepsilon} m$. *Let* p^* *be the smallest value such that the number of edges in* H'_{p^*} *is at least* $\frac{24n\delta\log(1/\varepsilon)\log n}{(1-\varepsilon)\varepsilon^3}$. *Notice that* p^* *is a function of the randomness in the hash function. Remark that the number of edges in* H'_{p^*} *is at most* $\frac{24n\delta\log(1/\varepsilon)\log n}{(1-\varepsilon)\varepsilon^3} + n \in \tilde{O}(n)$. *We denote* H'_{p^*} *by* $H_{\le n}(k,\varepsilon,\delta'')$, *and drop the parameters from* $H_{\le n}(k,\varepsilon,\delta'')$ *when it is clear from the context. See Algorithm 1.*

We argue that, for sufficiently large p, the quantity $\frac{1}{p}|\Gamma(H_p, S)|$ is a good estimate for $\mathcal{C}(S)$. This is formalized below.

Lemma 2.2. *Pick* $\frac{6\delta'}{\varepsilon^2 Opt_k} \le p \le 1$, *and let* S *be an arbitrarily subset of* $\mathcal{S}$ *such that* $|S| \le k$. *With probability* $1 - e^{-\delta'}$

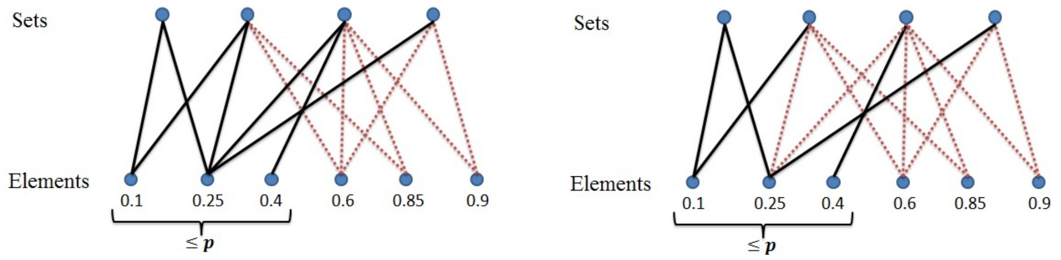

Figure 1: Left figure is an example of H_p and the right figure is an example of H'_p. In both figures, we have $p = 0.5$. The number below each vertex is its hashed value. Solid edges are those included in the sketch and dotted edges are the rest of the edges in the graph.

Algorithm 1 $H_{\leq n}(k, \varepsilon, \delta'')$

```
Input: An input graph $G$, $k$, $\varepsilon \in (0,1]$, and $\delta''$.
Output: Sketch $H_{\leq n}(k, \varepsilon, \delta'')$.
1: Set $\delta = \delta'' \log\log_{1-\varepsilon} m$.
2: Let $h$ be an arbitrary hash function that uniformly and independently maps $\mathcal{E}$ in $G$ to $[0,1]$.
3: Initialize $H_{\leq n}(k, \varepsilon, \delta'')$ with vertices $\mathcal{S}$ of $G$, and no edge.
4: while number of edges in $H_{\leq n}(k, \varepsilon, \delta'')$ is less than $\frac{24n\delta\log(1/\varepsilon)\log n}{(1-\varepsilon)\varepsilon^3}$ do
5:   Pick $v \in \mathcal{E}$ of minimum $h(v)$ that is still not in $H_{\leq n}(k, \varepsilon, \delta'')$.
6:   if degree of $v$ in $G$ is less than $\frac{n\log(1/\varepsilon)}{\varepsilon k}$ then
7:     Add $v$ along with all its edges to $H_{\leq n}(k, \varepsilon, \delta'')$.
8:   else
9:     Add $v$ along with $\frac{n\log(1/\varepsilon)}{\varepsilon k}$ of its edges, chosen arbitrary, to $H_{\leq n}(k, \varepsilon, \delta'')$.
```

we have

$$\left|\frac{1}{p}|\Gamma(H_p, S)| - \mathcal{C}(S)\right| \leq \varepsilon \mathsf{Opt}_k. \tag{1}$$

In the following lemma we relate the approximate solutions on H_p and G.

Lemma 2.3. *Pick $\frac{6k\delta\log n}{\varepsilon^2 \mathsf{Opt}_k} \leq p \leq 1$. All α-approximate solutions on H_p are $(\alpha - 2\varepsilon)$-approximate solutions to the k-cover problem on G with probability $1 - e^{-\delta}$. Simultaneously for any set $S \subseteq \mathcal{S}$ such that $|S| = k$, we have $\left|\frac{1}{p}|\Gamma(H_p, S)| - \mathcal{C}(S)\right| \leq \varepsilon \mathsf{Opt}_k$.*

Proof. Set $\delta' = k\delta\log n$. Lemma 2.2 states that for an arbitrary $S \subseteq \mathcal{S}$ of size at most k, we have with probability $1 - e^{-k\delta\log n}$,

$$\left|\frac{1}{p}|\Gamma(H_p, S)| - \mathcal{C}(S)\right| \leq \varepsilon \mathsf{Opt}_k.$$

Note that there are $\binom{n}{k}$ different sets S of size k. By the union bound, with probability $1 - \binom{n}{k}e^{-k\delta\log n} \geq 1 - n^k e^{-k\delta\log n} = 1 - e^{-\delta}$, we have for all such choices

$$\left|\frac{1}{p}|\Gamma(H_p, S)| - \mathcal{C}(S)\right| \leq \varepsilon \mathsf{Opt}_k. \tag{2}$$

Let Opt_k be the optimum solution on G and let S be the solution obtained from the α-approximation algorithm Alg when run on H_p. Applying Inequality (2) to Opt_k and S, we simultaneously have

$$\left|\frac{1}{p}|\Gamma(H_p, \mathsf{Opt}_k)| - \mathsf{Opt}_k\right| \leq \varepsilon \mathsf{Opt}_k \tag{3}$$

and

$$\left|\frac{1}{p}|\Gamma(H_p, S)| - \mathcal{C}(S)\right| \leq \varepsilon \mathsf{Opt}_k. \tag{4}$$

In addition, since S is an α-approximate solution on H_p we have

$$|\Gamma(H_p, \mathsf{Opt}_k)| \leq \frac{1}{\alpha}|\Gamma(H_p, S)|. \tag{5}$$

Inequalities (5) and (3) together ensure with probability $1 - e^{-\delta}$ that

$$\alpha \mathsf{Opt}_k - \frac{1}{p}|\Gamma(H_p, S)| \leq \alpha\varepsilon \mathsf{Opt}_k.$$

Combining the above with Inequality (4), we obtain

$$\alpha \mathsf{Opt}_k - \mathcal{C}(S) \leq \alpha\varepsilon \mathsf{Opt}_k + \varepsilon \mathsf{Opt}_k \leq 2\varepsilon \mathsf{Opt}_k.$$

This means that S is an $(\alpha - 2\varepsilon)$-approximation to k-cover on G as desired.

The following lemma relates the solutions on H'_p and H_p.

Lemma 2.4. *Pick arbitrary $0 \leq p \leq 1$ and $1 \leq k \leq n$. Any α-approximate solution of k-cover on H'_p is an $\alpha(1 - \varepsilon)$-approximate solution on H_p.*

Proof. Let Opt_H and $\mathsf{Opt}_{H'}$ be subsets of $\mathcal{S}$ with size k that maximize $|\Gamma(H_p, \mathsf{Opt}_H)|$ and $|\Gamma(H'_p, \mathsf{Opt}_{H'})|$, respectively. Remark that H'_p is a subgraph of H_p, hence $|\Gamma(H'_p, S)| \leq |\Gamma(H_p, S)|$ for any $S \subseteq \mathcal{S}$. Later we show that there exists a set R of size k such that $|\Gamma(H'_p, R)| \geq (1 - \varepsilon)|\Gamma(H_p, \mathsf{Opt}_H)|$. Thus, for an α-approximate solution

S on H'_p, we have

$$\begin{aligned}|\Gamma(H_p,S)| &\geq |\Gamma(H'_p,S)| && H'_p \subseteq H_p,\\ &\geq \alpha|\Gamma(H'_p,\mathsf{Opt}_{H'})| && S \text{ is } \alpha\text{-approximate},\\ &\geq \alpha|\Gamma(H'_p,R)| && \text{definition of } \mathsf{Opt}_{H'},\\ &\geq \alpha(1-\varepsilon)|\Gamma(H_p,\mathsf{Opt}_H)|.\end{aligned}$$

To prove the existence of a suitable R, we follow a probabilistic argument, producing a randomized set R^* of size k such that $\mathbf{E}[|\Gamma(H'_p,R^*)|] \geq (1-\varepsilon)|\Gamma(H_p,\mathsf{Opt}_H)|$.

We construct R^* by removing εk sets from Opt_H uniformly at random, and adding εk sets from $\mathcal{S}$ uniformly at random. Note that each element in $\Gamma(H_p,\mathsf{Opt}_H)$ with degree at most $\frac{n\log(1/\varepsilon)}{\varepsilon k}$ in H_p appears in $\Gamma(H_p,R^*)$ with probability $1-\varepsilon$, hence in $\Gamma(H'_p,R^*)$. Now let us consider a high-degree element u—one with degree at least $\frac{n\log(1/\varepsilon)}{\varepsilon k}$ in H_p, i.e., degree exactly $\frac{n\log(1/\varepsilon)}{\varepsilon k}$ in H'_p. The probability that u is not included in any of the εk randomly added sets is at most

$$\begin{aligned}\left(1-\frac{\frac{n\log(1/\varepsilon)}{\varepsilon k}}{n}\right)^{\varepsilon k} &= \left(1-\frac{\log(1/\varepsilon)}{\varepsilon k}\right)^{\varepsilon k}\\ &= \left(1-\frac{\log(1/\varepsilon)}{\varepsilon k}\right)^{\frac{\varepsilon k}{\log(1/\varepsilon)}\log\frac{1}{\varepsilon}}\\ &\leq \left(\frac{1}{e}\right)^{\log\frac{1}{\varepsilon}} = \varepsilon.\end{aligned}$$

Therefore, each vertex in $\Gamma(H_p,\mathsf{Opt}_H)$ exists in $\Gamma(H'_p,R^*)$ with probability at least $1-\varepsilon$, proving the claim $\mathbf{E}[|\Gamma(H'_p,R^*)|] \geq (1-\varepsilon)|\Gamma(H_p,\mathsf{Opt}_H)|$.

In the following two lemmas, we argue that maintaining the solution in the reduced-degree subgraph H'_p does not require too much memory.

Lemma 2.5. *Pick arbitrary $C \geq 1$ and let $p = \frac{6Ck\delta\log n}{\varepsilon^2 Opt_k}$. With probability at least $1-e^{1-\delta}$, we have*

$$\max_{S\subseteq\mathcal{S}:|S|=k} |\Gamma(H'_p,S)| \leq \frac{12Ck\delta\log n}{\varepsilon^2}.$$

Lemma 2.6. *Pick arbitrary $0 \leq p \leq 1$ and $1 \leq k \leq n$, and let m'_p denote the number of edges in H'_p. We have*

$$m'_p\frac{\varepsilon k}{2n\log(1/\varepsilon)} \leq |\Gamma(H'_p, Opt_{H'})|.$$

Proof. Let $\mathsf{Opt}_{H'} = \arg\max_S |\Gamma(H'_p,p)|$. There is a set $v \in \mathsf{Opt}_{H'}$ such that $|\Gamma(H'_p,\mathsf{Opt}_{H'})| - |\Gamma(H'_p,\mathsf{Opt}_{H'}-v)| \leq \frac{|\Gamma(H'_p,\mathsf{Opt}_{H'})|}{k}$, i.e., the marginal effect of v is at most a $\frac{1}{k}$ fraction of the total value of $|\Gamma(H'_p,\mathsf{Opt}_{H'})|$. Notice that by optimality of $\mathsf{Opt}_{H'}$, replacing v with any other set does not increase the union size. Thus, for any vertex $v' \in \mathcal{S}$ the number of neighbors of v' in $\mathcal{E}\setminus\Gamma(H'_p,\mathsf{Opt}_{H'})$ is at most $\frac{|\Gamma(H'_p,\mathsf{Opt}_{H'})|}{k}$. Therefore, the number of edges between $\mathcal{S}$ and $\mathcal{E}\setminus\Gamma(H'_p,\mathsf{Opt}_{H'})$ does not exceed $n\frac{|\Gamma(H'_p,\mathsf{Opt}_{H'})|}{k}$.

On the other hand, the degree of the elements in $\Gamma(H'_p,\mathsf{Opt}_{H'})$ is at most $\frac{\log(1/\varepsilon)n}{\varepsilon k}$, hence the number of edges between $\mathcal{S}$ and $\Gamma(H'_p,\mathsf{Opt}_{H'})$ does not exceed $|\Gamma(H'_p,\mathsf{Opt}_{H'})|\frac{n\log(1/\varepsilon)}{\varepsilon k}$. Therefore, one can bound the total number of edges in H'_p as follows.

$$\begin{aligned}m'_p &\leq n\frac{|\Gamma(H'_p,\mathsf{Opt}_{H'})|}{k} + |\Gamma(H'_p,\mathsf{Opt}_{H'})|\frac{n\log(1/\varepsilon)}{\varepsilon k}\\ &\leq \left|\Gamma(H'_p,\mathsf{Opt}_{H'})\right|\cdot\frac{n}{k}\left(1+\frac{\log(1/\varepsilon)}{\varepsilon}\right)\\ &\leq \left|\Gamma(H'_p,\mathsf{Opt}_{H'})\right|\cdot\frac{n}{k}\cdot\frac{2\log(1/\varepsilon)}{\varepsilon}.\end{aligned}$$

We obtain by reordering

$$m'_p\frac{\varepsilon k}{2n\log(1/\varepsilon)} \leq |\Gamma(H'_p,\mathsf{Opt}_{H'})|.$$

The following theorem relates the approximate solutions on $H_{\leq n}$ and G.

Theorem 2.7. *Let $\delta'' \in [1,\infty)$ and $k \in [1,n]$ be two arbitrary numbers. Any α-approximate solution to k-cover on $H_{\leq n}$ is an $\alpha - 12\epsilon$ approximation solution on G, with probability $1-3e^{-\delta''}$.*

Proof. Pick $p \geq \frac{6k\delta\log n}{\varepsilon^2\mathsf{Opt}_k}$. By Lemma 2.4, any α-approximate solution on H'_p is an $(\alpha-\varepsilon)$-approximate solution on H_p with probability $1-e^{-\delta}$. Moreover, we know from Lemma 2.3 that any $(\alpha-\varepsilon)$-approximate solution on H_p is an $(\alpha-3\varepsilon)$-approximate solution on G with probability $1-e^{-\delta}$. Therefore, any α-approximate solution on H'_p is an $(\alpha-3\varepsilon)$-approximate solution on G with probability $1-2e^{-\delta}$.

Let us set $p' = \frac{6k\delta\log n}{\varepsilon^2\mathsf{Opt}_k}$ and $p_0 = \frac{1}{m}, p_1 = \frac{1}{m(1-\varepsilon)}, p_2 = \frac{1}{m(1-\varepsilon)^2}, \ldots, p_\mu = 1$, where $\mu = O(\log m)$. Indeed, there is some i such that $p' \leq p_i \leq \frac{1}{1-\varepsilon}p'$. Remark that we set $\delta = \log(\log_{1-\varepsilon} m)\delta''$. We may assume without loss of generality that Lemmas 2.3, 2.4 and 2.5 all hold for every p_j with $j \geq i$ since union bound ensures this outcome happens with probability at least

$$\begin{aligned}1-3\log_{1-\varepsilon} m\cdot e^{\delta} &= 1-3\log_{1-\varepsilon} m\exp[\log(\log_{1-\varepsilon} m)\delta'']\\ &= 1-3e^{\delta''}.\end{aligned}$$

Let p^* be (a random number) such that $p^* \geq \frac{1}{1-\varepsilon}\frac{6k\delta\log n}{\varepsilon^2\mathsf{Opt}_k}$. Remark that, since $p^* \geq \frac{1}{1-\varepsilon}p'$, there is some (random number) j such that $p' \leq p_j \leq p^* \leq p_{j+1} = \frac{p_j}{1-\varepsilon}$. Thus,

$$\left|\mathsf{Opt}_k - \frac{1}{p_j}\mathsf{Opt}(H'_{p_j})\right| \leq 3\varepsilon\mathsf{Opt}_k, \tag{6}$$

and similarly,

$$\begin{aligned}3\varepsilon\mathsf{Opt}_k &\geq \left|\mathsf{Opt}_k - \frac{1}{p_{j+1}}\mathsf{Opt}(H'_{p_{j+1}})\right|\\ &= \left|\mathsf{Opt}_k - \frac{1-\varepsilon}{p_j}\mathsf{Opt}(H'_{p_{j+1}})\right|\\ &\geq (1-\varepsilon)\left|\mathsf{Opt}_k - \frac{1}{p_j}\mathsf{Opt}(H'_{p_{j+1}})\right| - \varepsilon\mathsf{Opt}_k,\end{aligned}$$

which, assuming $\varepsilon \leq \frac{1}{5}$, gives

$$5\varepsilon\mathsf{Opt}_k \geq \left|\mathsf{Opt}_k - \frac{1}{p_j}\mathsf{Opt}(H'_{p_{j+1}})\right|. \tag{7}$$

Combining (6) and (7) yields

$$\left|\frac{1}{p_j}\mathsf{Opt}(H'_{p_{j+1}}) - \frac{1}{p_j}\mathsf{Opt}(H'_{p_j})\right| \leq 8\varepsilon\mathsf{Opt}_k. \tag{8}$$

The inequalities $p_j \leq p^* \leq p_{j+1}$ implies $H'_{p_j} \subseteq H'_{p^*} \subseteq H'_{p_{j+1}}$, hence

$$\Gamma(H'_{p_j}, S) \leq \Gamma(H'_{p^*}, S) \leq \Gamma(H'_{p_{j+1}}, S) \qquad \text{for any set } S, \tag{9}$$

and in turn,

$$\mathsf{Opt}(H'_{p_j}) \leq \mathsf{Opt}(H'_{p^*}) \leq \mathsf{Opt}(H'_{p_{j+1}}). \tag{10}$$

Combining (8) and (10) gives

$$\frac{1}{p_j}\mathsf{Opt}(H'_{p_{j+1}}) - \frac{1}{p_j}\mathsf{Opt}(H'_{p^*}) \leq 8\varepsilon\mathsf{Opt}_k. \tag{11}$$

Now suppose S is an α-approximate solution on H'_{p^*}. We have

$$\begin{aligned}
&\mathcal{C}(S) + \varepsilon\mathsf{Opt}_k \\
&\geq \frac{1}{p_{j+1}}|\Gamma(H_{p_{j+1}}, S)| && \text{Lemma 2.3,} \\
&\geq \frac{1}{p_{j+1}}|\Gamma(H'_{p_{j+1}}, S)| && H'_{p_{j+1}} \subseteq H_{p_{j+1}}, \\
&\geq \frac{1}{p_{j+1}}|\Gamma(H'_{p^*}, S)| && \text{by (9),} \\
&\geq \alpha\frac{1}{p_{j+1}}\mathsf{Opt}(H'_{p^*}) && \text{definition of } S, \\
&= \alpha(1-\varepsilon)\frac{1}{p_j}\mathsf{Opt}(H'_{p^*}) \\
&\geq \alpha(1-\varepsilon)[\frac{1}{p_j}\mathsf{Opt}(H'_{p_{j+1}}) - 8\varepsilon\mathsf{Opt}_k] && \text{from (11),} \\
&\geq \alpha\frac{1}{p_{j+1}}\mathsf{Opt}(H'_{p_{j+1}}) - \alpha 8\varepsilon(1-\varepsilon)\mathsf{Opt}_k \\
&\geq \alpha\frac{1}{p_{j+1}}\mathsf{Opt}(H'_{p_{j+1}}) - \alpha 8\varepsilon\mathsf{Opt}_k \\
&\geq \alpha\frac{1}{p_{j+1}}\mathsf{Opt}(H'_{p_{j+1}}) - 8\varepsilon\mathsf{Opt}_k && \text{since } \alpha \leq 1, \\
&\geq \alpha\mathsf{Opt}_k - 11\varepsilon\mathsf{Opt}_k \\
&= (\alpha - 11\varepsilon)\mathsf{Opt}_k,
\end{aligned}$$

that is, any α-approximate solution on H'_{p^*} is an $(\alpha - 12\varepsilon)$-approximate solution on G.

Finally we argue that $p^* \geq \frac{1}{1-\varepsilon}\frac{6k\delta\log n}{\varepsilon^2\mathsf{Opt}_k}$ for $H'_{p^*} = H_{\leq n}$. We set $C = \frac{1}{1-\varepsilon}$ in Lemma 2.5 to obtain $\max_{S\subseteq\mathcal{S}:|S|=k}|\Gamma(H'_{p''}, S)| \leq \frac{\frac{12k\delta}{1-\varepsilon}\log n}{\varepsilon^2}$, where $p'' = \frac{1}{1-\varepsilon}\frac{6k\delta\log n}{\varepsilon^2\mathsf{Opt}_k}$. Thus, H'_{p^*} contains $H'_{p''}$ if $\max_{S\subseteq\mathcal{S}:|S|=k}|\Gamma(H'_{p^*}, S)| \geq \frac{\frac{12k\delta}{1-\varepsilon}\log n}{\varepsilon^2}$. On the other hand if we set $m'_{p^*} \geq \frac{24n\delta\log(1/\varepsilon)\log n}{(1-\varepsilon)\varepsilon^3}$ in Lemma 2.6, we get

$$\begin{aligned}
|\Gamma(H'_{p^*}, \mathsf{Opt}_{H'})| &\geq m'_{p^*}\frac{\varepsilon k}{2n\log(1/\varepsilon)} \\
&\geq \frac{24n\delta\log(1/\varepsilon)\log n}{(1-\varepsilon)\varepsilon^3}\cdot\frac{\varepsilon k}{2n\log(1/\varepsilon)} \\
&= \frac{\frac{12k\delta}{1-\varepsilon}\log n}{\varepsilon^2}.
\end{aligned}$$

The following lemma provides a bicriteria bound on the coverage of solutions in $H_{\leq n}$, where $k' \leq k$ is the size of the set cover on G. This lemma will be useful in obtaining results for set cover and set cover with outliers.

Lemma 2.8. *Let k' be the size of the minimum set cover on the input graph G, and let $k = \xi k'$. There exists a solution of size k' on $H_{\leq n}(k, \varepsilon, 1)$ that covers at least $1 - \xi\varepsilon$ fraction of the elements in $H_{\leq n}(k, \varepsilon, 1)$.*

3. THE STREAMING SETTING

Indeed, with no time constraint, one can use ℓ_0 sketches and give a $1-\varepsilon$ approximation streaming algorithm for k-cover in $\tilde{O}(nk)$ space; see Appendix C. This simple streaming algorithm constructs a $(1\pm\varepsilon)$-approximate oracle to the value of the coverage function, using $\tilde{O}(nk)$ space. One can use this algorithm and try all solutions of size k to find a $1-2\varepsilon$ approximate solution of k-cover. However, as Theorem 1.3 states, using this oracle and without any further assumptions, there is no polynomial time $n^{-0.5+\varepsilon}$-approximation algorithm for k-cover. In addition, the space used by this algorithm may be quite large for large values of k.

In this section, we improve the algorithm provided in Appendix C and give a $1-\frac{1}{e}-\varepsilon$-approximation one-pass streaming algorithm for k-cover, using $\tilde{O}(n)$ space. This is done by first constructing $H_{\leq n}$ in the streaming setting and then providing efficient algorithms that only access the sketch $H_{\leq n}$. Using the same technique, we give a $(1+\varepsilon)\log\frac{1}{\lambda}$ approximation one-pass streaming algorithm for set cover with outliers, using $\tilde{O}_\lambda(n)$ space. Besides, for any arbitrary $r \in [1, \log m]$, we give a $(1+\varepsilon)\log m$ approximation r-pass streaming algorithm for set cover, using $\tilde{O}(nm^{O(1/r)} + m)$ space. Interestingly, the update times of all our algorithms are $\tilde{O}(1)$.

On the hardness side we show in Appendix 4 that any $\frac{1}{2}+\varepsilon$ approximation streaming algorithm for the k-cover problem requires $\Omega(n)$ space. This rules out the existence of $\frac{1}{2}+\varepsilon$ approximation parametrized streaming algorithms for the k-cover problem and shows that the space of our algorithm is tight up to a logarithmic factor.

Next we show how to construct $H_{\leq n}$ in the streaming setting. Note that to define $H_{\leq n}$ we map (via a hash function) each element to a number in $[0,1]$ independently. Such a mapping requires $\tilde{O}(m)$ random bits. However, we use a simple equivalent random process to construct $H_{\leq n}$ using $\tilde{O}(|H_{\leq n}|)$ random bits, where $|H_{\leq n}|$ is the number of edges in $H_{\leq n}$.

Let p^* be the probability corresponding to $H_{\leq n}$. Remark that the number of elements v with $h(v) \leq p^*$ is at most $|H_{\leq n}|$, i.e., at most equal to the number of edges in $H_{\leq n}$. Note that, if we know that the hash value of an element is greater than p^*, we can simply remove that element. Thus, at the beginning we iteratively sample $|H_{\leq n}|$ elements without replacement, and assume that this sequence is indeed that of the first $|H_{\leq n}|$ elements ordered by their hashed value. This process requires only $\tilde{O}(|H_{\leq n}|)$ random bits.

Next we describe how to use the sketch to solve each of the three problems: k-cover, set cover, and set cover with outliers. As a result, we provide tight and almost tight streaming algorithms for k-cover, set cover, and set cover with outliers.

The greedy algorithm for k-cover iteratively selects a vertex that increases the valuation function f the most and adds it to the solution. Let $\mathsf{Greedy}(k, G)$ denote the set of k vertices picked by the greedy algorithm when run on input graph G. It is known that the Greedy is a $1-\frac{1}{e}$

Algorithm 2 Streaming algorithm to compute $H_{\leq n}(k, \varepsilon, \delta'')$

```
Input: An input graph $G$, $k$, $\varepsilon \in (0,1]$, and $\delta''$.
Output: Sketch $H_{\leq n}(k, \varepsilon, \delta'')$.
Initialization:
 1: Set $\delta = \delta'' \log\log_{1-\varepsilon} m$.
 2: Pick $\frac{24n\delta \log(1/\varepsilon)\log n}{(1-\varepsilon)\varepsilon^3} + \frac{n\log(1/\varepsilon)}{\varepsilon k}$ element from $\mathcal{E}$ uniformly at random and let $\Pi$ be a random permutation over these elements.
 3: Initialize $H_{\leq n}(k, \varepsilon, \delta'')$ with vertices $\mathcal{S}$ of $G$, and no edge.
Update edge $(u, v)$:
 1: if $v$ is not sampled in $\Pi$ then
 2:     Discard $(u, v)$.
 3: else if degree of $v$ in $G$ is $\frac{n\log(1/\varepsilon)}{\varepsilon k}$ then
 4:     Discard $(u, v)$.
 5: else
 6:     Add $(u, v)$ to $H_{\leq n}(k, \varepsilon, \delta'')$.
 7: while number of edges in $H_{\leq n}(k, \varepsilon, \delta'')$ is more than $\frac{24n\delta \log(1/\varepsilon)\log n}{(1-\varepsilon)\varepsilon^3} + \frac{n\log(1/\varepsilon)}{\varepsilon k}$ do
 8:     Let $w$ be the last element in $\Pi$.
 9:     Remove $w$ from $\Pi$.
10:     Remove $w$ from $H_{\leq n}(k, \varepsilon, \delta'')$.
```

approximation algorithm [40]. In addition, we know that $\mathcal{C}(\mathsf{Greedy}(k\log\frac{1}{\lambda}, G)) \leq (1-\lambda)\mathsf{Opt}_k(G)$.

Theorem 3.1. *For any $\varepsilon \in (0,1]$ and any graph G, Algorithm 3 produces a $(1-\frac{1}{e}-\varepsilon)$-approximate solution to k-cover on G with probability $1-\frac{1}{n}$. The number of edges in the sketch used by this algorithm is $\tilde{O}(n)$.*

Lemma 3.2. *For arbitrary k', $\varepsilon' \in (0,1]$, $\lambda' \in (0, \frac{1}{e}]$, $C' \in [1,\infty)$, and graph G, Algorithm 4 returns false only if the size of the minimum set cover of G is greater than k'. Otherwise, the algorithm returns a solution of size $k'\log\frac{1}{\lambda'}$ that covers $1-\lambda'-\varepsilon'$ fraction of $\mathcal{E}$ in G with probability $1-\frac{1}{C'n}$. The number of edges in the sketch used by this algorithm is $O(\frac{n\log^2 n\log^6 m\log C'}{\varepsilon'^3})$.*

Theorem 3.3. *Given $\varepsilon \in (0,1]$, $C \geq 1$ and a graph G, Algorithm 5 returns a $(1+\varepsilon)\log\frac{1}{\lambda}$ approximate solution to set cover with λ outliers on G with probability $1-\frac{1}{n}$. The total number of edges in the sketches used by this algorithm is $\tilde{O}(n/\lambda^3) \subseteq \tilde{O}_\lambda(n)$.*

Proof. Remark that each iteration of the loop in Algorithm 5 increases k' by a factor of $1+\frac{\varepsilon}{3}$, and we always keep $k' \leq n$. Thus, we run at most $\log_{1+\frac{\varepsilon}{3}} n$ instances of Algorithm 4. Lemma 3.2 holds for each of these instances with probability $\frac{1}{Cn\log_{1+\frac{\varepsilon}{3}} n}$. They all hold with probability $1-\frac{1}{Cn}$ by the union bound. We prove the statement of the theorem assuming this.

Let k^* be the size of the minimum set cover in G, and let k' be the final value of this variable after run of Algorithm 5. Indeed Algorithm 4 returns false for $\frac{k'}{1+\varepsilon/3}$, hence $k' \leq (1+\varepsilon/3)k^*$. Note that, the size of the set returned by Algorithm 5 is

$$\begin{aligned}
k'\log\frac{1}{\lambda'} &= k'\log\frac{1}{\lambda e^{-\varepsilon/2}} \\
&= k'\left[\log\frac{1}{\lambda} + \frac{\varepsilon}{2}\right] \\
&\leq \left[\log\frac{1}{\lambda} + \frac{\varepsilon}{2}\right]\cdot\left[1+\frac{\varepsilon}{3}\right]\cdot k^* \\
&= k^*\left[\log\frac{1}{\lambda} + \frac{\varepsilon}{2} + \frac{\varepsilon}{3}\log\frac{1}{\lambda} + \frac{\varepsilon^2}{6}\right] \\
&\leq (1+\varepsilon)k^*\log\frac{1}{\lambda}.
\end{aligned}$$

On the other hand, this solution covers at least

$$1-\lambda'-\varepsilon' = 1-\lambda e^{-\varepsilon/2} - \lambda(1-e^{-\varepsilon/2}) = 1-\lambda$$

fraction of the vertices in G, as claimed.

Lemma 3.2 bounds the number of edges in the sketch used by each instance of Algorithm 4 as

$$\begin{aligned}
&O\left(\frac{n\log^2 n\log^6 m\log C'}{\varepsilon'^3}\right) \\
&= O\left(\frac{n\log^2 n\log^6 m\log C\log_{1+\frac{\varepsilon}{3}} n}{[\lambda(1-e^{-\varepsilon/2})]^3}\right) \\
&\subseteq \tilde{O}(n/\lambda^3) \subseteq \tilde{O}_\lambda(n).
\end{aligned}$$

With $\log_{1+\varepsilon/3} n$ runs of Algorithm 4, the total number of edges in all the sketches used in this algorithm is $\tilde{O}(n/\lambda^3) \subseteq \tilde{O}_\lambda(n)$.

We implement each iteration of Algorithm 6 in two streaming passes. In the first pass of each iteration we simply mark covered elements to virtually construct G_i, whereas in the second pass, we construct $H_{\leq n}$. After all $r-1$ iterations, we utilize one extra pass to keep all edges to construct G_r. Hence, the following theorem proves the third statement of Theorem 1.1

Theorem 3.4. *Given $\varepsilon \in (0,1]$ and a graph G, Algorithm 6 finds a $(1+\varepsilon)\log m$ approximate solution to set cover on G*

Algorithm 3 k-cover

```
Input: An input graph $G$, $k$, and $\varepsilon \in (0,1]$.
Output: A $1-\frac{1}{e}-\varepsilon$ approximate solution to $k$-cover on $G$ with probability $1-\frac{1}{n}$.
1: Set $\delta'' = 2 + \log n$ and $\varepsilon' = \frac{1}{12}\varepsilon$.
2: Construct sketch $H_{\leq n}(k, \varepsilon', \delta'')$.    // Compute this over the stream.
3: Run the greedy algorithm (or any $1-\frac{1}{e}$ approximation algorithm) on this sketch and report $\mathsf{Greedy}(k, H_{\leq n}(k, \varepsilon', \delta''))$.
```

Algorithm 4 A submodule to solve set cover

```
Input: Parameters $k'$, $\varepsilon' \in (0,1]$, $\lambda' \in (0, \frac{1}{e}]$, and $C' \in [1, \infty)$, as well as a graph $G$ promised to have a set cover of size $k'$.
Output: A solution of size $k' \log \frac{1}{\lambda'}$ covering $1-\lambda'-\varepsilon'$ fraction of $\mathcal{E}$ in $G$ with probability $1-\frac{1}{C'n}$.
1: Set $\delta'' = \log_{1+\varepsilon} n[\log(C'n)+2]$ and $\varepsilon = \frac{\varepsilon'}{13\log\frac{1}{\lambda'}}$.
2: Construct sketch $H_{\leq n}(k' \log \frac{1}{\lambda'}, \varepsilon, \delta'')$.    // Compute this over the stream.
3: Run the greedy algorithm on this sketch to get solution $S = \mathsf{Greedy}(k' \log \frac{1}{\lambda'}, H_{\leq n})$
4: if $S$ covers at least $1-\lambda'-\varepsilon\log\frac{1}{\lambda'}$ fraction of $\mathcal{E}$ in $H_{\leq n}$ then
5:     return $S$
6: else
7:     return false
```

Algorithm 5 Set cover with λ outliers

```
Input: A graph $G$ and parameters $\varepsilon \in [0,1]$, $\lambda \in (0, \frac{1}{e}]$, and $C \geq 1$.
Output: A $(1+\varepsilon)\log\frac{1}{\lambda}$ approximate solution to set cover with $\lambda$ outliers on $G$ with probability $1-\frac{1}{Cn}$.
1: Set $\varepsilon' = \lambda(1-e^{-\varepsilon/2})$, and $\lambda' = \lambda e^{-\varepsilon/2}$, and $C' = C\log_{1+\frac{\varepsilon}{3}} n$, and $k' = 1$.
2: repeat
3:     $k' \leftarrow (1+\frac{\varepsilon}{3})k'$
4:     Run Algorithm 4 on $(k', \varepsilon', \lambda', C', G)$ and let $S$ be the outcome.    // Run these in parallel.
5: until $S$ is not false or $k' = n$
6: return $S$
```

Algorithm 6 Set cover in r iterations

```
Input: A graph $G$ as well as $\varepsilon \in (0,1]$, $C \geq 1$, and $r \in [1, \log m]$.
Output: A $(1+\varepsilon)\log m$ approximate solution to set cover of $G$ with probability $1-\frac{1}{Cn}$.
1: Let $G_1 = G$, $\lambda = m^{-\frac{1}{2+r}}$, $C' = (r-1)C$, $S = \emptyset$.
2: for $i = 1$ to $r-1$ do
3:     Run Algorithm 5 on $(G_i, \varepsilon, \lambda, C')$ and let $S_i$ to be the outcome.    // i-th streaming pass.
4:     Add $S_i$ to $S$
5:     Remove from $G_i$ the elements covered by $S_i$ and call the new graph $G_{i+1}$.
6: Run the greedy algorithm to find a set cover of $G_r$ and let $S^{\mathsf{Greedy}}$ to be the result.
7: Add $S^{\mathsf{Greedy}}$ to $S$.
8: return $S$
```

with probability $1 - \frac{1}{n}$. The total number of edges in the sketches used by this algorithm plus the number of edges in G_r is at most $\tilde{O}(nm^{\frac{3}{2+r}}) \subseteq \tilde{O}(nm^{O(1/r)})$.

Proof. The algorithm runs $r - 1$ instances of Algorithm 5. Theorem 3.3 holds for each with probability $1 - \frac{1}{C'n} = 1 - \frac{1}{(r-1)Cn}$, hence for all simultaneously with probability $1 - (r-1)\frac{1}{(r-1)Cn} = 1 - \frac{1}{Cn}$. We assume these hold when proving the statement of the theorem.

Let k' be the size of the minimum set cover in G. Note that for any $i \in [1, r]$, G_i is an induced subgraph of G that contains all sets in G. Thus, any set cover of G is a set cover of G_i as well. This means that the size of the set cover of G_i is at most k'. Therefore, Theorem 3.3 bounds the number of sets chosen by each run of Algorithm 5 by $(1+\varepsilon)\log\frac{1}{\lambda}k' = (1+\varepsilon)\log m^{\frac{1}{2+r}}k'$. Also, each run of Algorithm 5 covers $1-\lambda$ fraction of the remaining uncovered elements. Therefore, the number of uncovered elements in G_i is at most $m\lambda^{i-1}$, and in particular this is $m\lambda^{r-1} = m^{\frac{3}{2+r}}$ for G_r. Therefore, the total size of the set cover obtained by this algorithm is at most

$$\begin{aligned}
&(r-1)(1+\varepsilon)k'\log m^{\frac{1}{2+r}} + k'\log m^{\frac{3}{2+r}} \\
&\le (1+\varepsilon)k'\left[(r-1)\log m^{\frac{1}{2+r}} + \log m^{\frac{3}{2+r}}\right] \\
&= (1+\varepsilon)k'\left[(r-1)\frac{1}{2+r} + \frac{3}{2+r}\right]\log m \\
&= (1+\varepsilon)k'\log m.
\end{aligned}$$

Remark that the total number of edges in the sketches used by Algorithm 5 is $\tilde{O}(n/\lambda^3)$. With $r \le \log m$ such runs, the total number of edges in all the sketches is $\tilde{O}(n/\lambda^3) = \tilde{O}(nm^{\frac{3}{2+r}})$. On the other hand, the number uncovered elements in G_r is $m\lambda^{r-1} = m^{\frac{3}{2+r}}$. Thus, the number of edges in G_r is at most $nm^{\frac{3}{2+r}}$. Therefore, the total number of edges in the sketches plus the number of edges in G_r is $\tilde{O}(nm^{\frac{3}{2+r}}) \subseteq \tilde{O}(nm^{O(1/r)})$.

4. HARDNESS OF STREAMING PROBLEMS

Here, we give a lower bound on the space required to solve k-cover in the streaming setting. To establish this lower bound, we present a reduction from the set-disjointness problem. In the set disjointness problem, two parties, namely Alice and Bob, each holds a subset of $1, 2, \ldots, n$. The goal is to determine whether the sets are disjoint or not. Razborov [43] and Kalyanasundaram and Schintger [29] provide a lower bound of $\Omega(n)$ even when allowing randomization.

Proof Proof of Theorem 1.2. Let A be the set that Alice holds, and let B be the set that Bob holds. In our hard k cover instance we have two vertices (namely a and b) in $\mathcal{E}$ and n vertices in $\mathcal{S}$. Vertex a has an edge to the i-th vertex in $\mathcal{E}$ if and only if i exists in A. Similarly, b has an edge to the i-th vertex in $\mathcal{E}$ if and only if i exists in B. In the input stream first we see the edges of a (which contains the information Alice holds) and then the edges of b (which contains the information Bob holds).

In this example, if the sets A and B are disjoint, each of the vertices in $\mathcal{S}$ covers at most one of a and b, and thus, the value of an optimum solution to 1-cover on this graph is 1. Otherwise, there is a vertex i which has edge to both a and b, and thus, the value of an optimum solution to 1-cover on this graph is 2. Therefore, distinguishing between the case that the value of the optimum solution to 1-cover is 1 and the case that this value is 2 requires $\Omega(n)$ space in total.

5. THE K-COVER PROBLEM VIA $(1 \pm \varepsilon)$-APPROXIMATE ORACLE

In this section we consider the approximability of k-cover using the $(1\pm\varepsilon)$-approximate oracle, and prove Theorem 1.3 by showing that any α-approximation algorithm via oracle $\mathcal{C}_\varepsilon$ requires at least $\exp\left(\Omega(n\varepsilon^2\alpha^2 - \log n)\right)$ oracle queries.

Theorem 5.2 states the hardness of the k-purification problem. Its proof uses the following generalization of the Chernoff bound.

Lemma 5.1. *Let X be the sum of several negatively correlated binary random variables. We have*

$$\mathbf{Pr}\left(|X - \mathbf{E}[X]| > \gamma\right) \le 2\exp\left(-\frac{\gamma^2}{3\mathbf{E}[X]}\right).$$

Proof. Panconesi and Srinivasan [42] show that if X is the sum of certain negatively correlated binary random variables, we have $\mathbf{Pr}\left(|X - \mathbf{E}[X]| > \varepsilon\mathbf{E}[X]\right) \le 2\exp\left(-\frac{\varepsilon^2\mathbf{E}[X]}{3}\right)$. Setting $\gamma = \varepsilon\mathbf{E}[X]$ yields

$$\begin{aligned}
\mathbf{Pr}\left(|X - \mathbf{E}[X]| > \gamma\right) &\le 2\exp\left(-\frac{\left(\frac{\gamma}{\mathbf{E}[X]}\right)^2 \cdot \mathbf{E}[X]}{3}\right) \\
&= 2\exp\left(-\frac{\gamma^2}{3\mathbf{E}[X]}\right)
\end{aligned}$$

as desired.

Theorem 5.2. *Any randomized algorithm that solves k-purification with probability at least δ requires at least $\delta\exp\left(\Omega(\frac{\varepsilon^2k^2}{n})\right)$ oracle queries.*

Proof. By Yao's principle we can restrict our analysis to deterministic algorithms. Let Alg be a deterministic algorithm for k-purification. Suppose that after q queries, algorithm Alg finds with probability δ a set S such that $\mathsf{Pure}_\varepsilon(S) = 1$. Let $S_1, S_2, \ldots, S_q$ be the q subsets queried by Alg. Definition of δ and the union bound give

$$\delta \le \sum_{i=1}^{q} \mathbf{Pr}\big(\mathsf{Pure}_\varepsilon(S_i) = 1\big). \tag{12}$$

Now, we provide an upper bound to $\mathbf{Pr}\big(\mathsf{Pure}_\varepsilon(S) = 1\big)$ for an arbitrary subset S. Let X_i be a random variable that indicates whether the i-th item in S is gold. Let $X = \sum_{i=1}^{|S|} X_i$. Indeed, X_i variables are negatively correlated [28].

We set $\gamma = \varepsilon(\frac{k|S|}{n} + \frac{k^2}{n})$ in Lemma 5.1 to obtain

$$\begin{aligned}\mathbf{Pr}\big(\mathsf{Pure}_\varepsilon(S) = 1\big) &= \mathbf{Pr}\left(|X - \mathbf{E}[X]| > \varepsilon\left(\frac{k|S|}{n} + \frac{k^2}{n}\right)\right)\\ &\le 2\exp\left(-\frac{\varepsilon^2(\frac{k|S|}{n} + \frac{k^2}{n})^2}{3\mathbf{E}[X]}\right)\\ &\le 2\exp\left(-\frac{\varepsilon^2(\frac{k|S|}{n} + \frac{k^2}{n})^2}{3\frac{k|S|}{n}}\right)\\ &\le 2\exp\left(-\frac{\varepsilon^2 k(|S|+k)^2}{3n|S|}\right)\\ &\le 2\exp\left(-\frac{\varepsilon^2k^2}{3n}\right).\end{aligned}$$

Coupled with Inequality (12) the above implies that $\delta \le \sum_{i=1}^{q} \mathbf{Pr}\big(\mathsf{Pure}_\varepsilon(S_i) = 1\big) \le 2q\exp\left(-\frac{\varepsilon^2k^2}{3n}\right)$, which means $q \ge \frac{\delta}{2}\exp\left(\frac{\varepsilon^2k^2}{3n}\right)$, as desired.

Proof Proof of Theorem 1.3. Given an instance of the k-purification problem we construct a k-cover instance with a $(1 \pm \varepsilon')$-approximate oracle as follows. We associate one set for each gold or brass item in the original instance in such a way that the value of the coverage function (for nonempty S) is $\mathcal{C}(S) = k + \frac{n}{k}\mathsf{Gold}(S)$; i.e., there are k elements common between all gold and brass *sets*, and in addition, each gold set contains $\frac{n}{k}$ additional exclusive elements. The optimum solution consists of all gold sets, hence

$$\mathsf{Opt} = k + \frac{n}{k}k = k + n > n. \tag{13}$$

We define

$$\mathcal{C}_{\varepsilon'}(S) = \begin{cases} k + |S| & \text{if } \mathsf{Pure}_\varepsilon(S) = 0\\ \mathcal{C}(S) & \text{otherwise.}\end{cases}$$

We claim that $\mathcal{C}_{\varepsilon'}$ is a $(1 \pm \varepsilon')$-approximate oracle to f for $\varepsilon' = 2\varepsilon$. We set $\varepsilon' = 2\varepsilon$. Notice that for $\mathsf{Pure}_\varepsilon(S) = 1$, the estimate $\mathcal{C}_{\varepsilon'}(S)$ is clearly within the $1 \pm \varepsilon'$ factor of $\mathcal{C}(S)$. Moreover when $\mathsf{Pure}_\varepsilon(S) = 0$, we have $\frac{k|S|}{n} - \varepsilon\big(\frac{k|S|}{n} + \frac{k^2}{n}\big) \le \mathsf{Gold}(S) \le \frac{k|S|}{n} + \varepsilon\big(\frac{k|S|}{n} + \frac{k^2}{n}\big)$. Thus we have

$$\begin{aligned}(1-\varepsilon')\mathcal{C}(S) &\le \frac{1}{1+\varepsilon}\mathcal{C}(S)\\ &= \frac{1}{1+\varepsilon}\left[k + \frac{n}{k}\mathsf{Gold}(S)\right]\\ &\le \frac{1}{1+\varepsilon}\left[k + \frac{n}{k}\left(\frac{k|S|}{n} + \varepsilon\left(\frac{k|S|}{n} + \frac{k^2}{n}\right)\right)\right]\\ &= \frac{1}{1+\varepsilon}\left[k + |S| + \varepsilon(|S| + k)\right]\\ &= k + |S| = \mathcal{C}_{\varepsilon'}(S).\end{aligned}$$

Similarly we have

$$\begin{aligned}(1+\varepsilon')\mathcal{C}(S) &\ge \frac{1}{1-\varepsilon}\mathcal{C}(S)\\ &= \frac{1}{1-\varepsilon}\left[k + \frac{n}{k}\mathsf{Gold}(S)\right]\\ &\ge \frac{1}{1-\varepsilon}\left[k + \frac{n}{k}\left(\frac{k|S|}{n} - \varepsilon\left(\frac{k|S|}{n} + \frac{k^2}{n}\right)\right)\right]\\ &= \frac{1}{1-\varepsilon}\left[k + |S| - \varepsilon(|S| + k)\right]\\ &= k + |S| = \mathcal{C}_{\varepsilon'}(S).\end{aligned}$$

Therefore, $\mathcal{C}_{\varepsilon'}$ is a $(1 \pm \varepsilon')$-approximate oracle to $\mathcal{C}$.

For an arbitrary subset S of size k with $\mathsf{Pure}_\varepsilon(S) = 0$, we have

$$\begin{aligned}\frac{\mathcal{C}(S)}{\mathsf{Opt}} &< \frac{k + \frac{n}{k}\mathsf{Gold}(S)}{n}\\ &\le \frac{k + \frac{n}{k}\left[\frac{k|S|}{n} + \varepsilon\left(\frac{k|S|}{n} + \frac{k^2}{n}\right)\right]}{n}\\ &\le \frac{k + \frac{n}{k}\left[\frac{k^2}{n} + \varepsilon\left(\frac{k^2}{n} + \frac{k^2}{n}\right)\right]}{n}\\ &= \frac{(2k + 2\varepsilon k)}{n}\\ &\le \frac{4k}{n}.\end{aligned}$$

Thus, if S is a $\frac{4k}{n}$-approximate solution to the k-cover instance, we have $\mathsf{Pure}_\varepsilon(S) = 1$. Therefore, any $\frac{6k}{n}$-approximation algorithm returns a set S such that $\mathsf{Pure}_\varepsilon(S) = 1$ with probability at least $\frac{6k/n - 4k/n}{\mathsf{Opt}} = \frac{2k/n}{n+k} = \frac{2k}{n^2+kn} \ge \frac{1}{n^2}$.

Recall that for any subset S given that $\mathsf{Pure}_\varepsilon(S) = 0$, the value of $\mathcal{C}_{\varepsilon'}(S)$ is predetermined, and can be computed independent of the actual value of $\mathcal{C}(S)$. Thus, using a $\frac{6k}{n}$-approximation algorithm for the k-cover problem with ε'-error oracle, with probability $\frac{1}{n^2}$, one can find a set S such that $\mathsf{Pure}_\varepsilon(S) = 1$, using the same number of queries. Theorem 5.2 states that the number of queries is not less than

$$\begin{aligned}\frac{1}{n^2}\exp\left(\Omega\Big(\frac{\varepsilon'^2k^2}{n}\Big)\right) &= \frac{1}{n^2}\exp\left(\Omega\Big(\frac{\varepsilon^2k^2}{n}\Big)\right)\\ &\in \exp\left(\Omega\Big(\frac{\varepsilon^2k^2}{n} - \log n\Big)\right)\\ &= \exp\left(\Omega\Big(n\varepsilon^2\alpha^2 - \log n\Big)\right).\end{aligned}$$

6. CONCLUSION

In this paper, we presented a simple, yet powerful sketching technique for coverage problems, and showed how to construct this sketch in streaming model. The streaming results improve the state of the art in three dimensions: approximation ratio, space complexity, and streaming arrival model (i.e., from set-arrival to element- or edge-arrival model). In an accompanied paper, we also applied this sketching idea for distributed computation models (such as MapReduce), and show how it improves the best known results in that area as well. More notably, we also performed an extensive empirical evaluation of resulting distributed algorithms and show the effectiveness of applying this sketching technique for analyzing massive data sets in practice [10]. As noted earlier, this sketch and the distributed and streaming algorithms based on it work very well in instances in which the size of the subsets is large. Notably, all the other techniques (e.g., based on composable core-sets) fail in these regimes. As future research, we hope this technique can be applied to other computation models and other problems.

7. REFERENCES

[1] Z. Abbassi, V. S. Mirrokni, and M. Thakur. Diversity maximization under matroid constraints. In *KDD*, pages 32–40, 2013.

[2] K. J. Ahn and S. Guha. Graph sparsification in the semi-streaming model. In *ICALP (2)*, pages 328–338, 2009.

[3] K. J. Ahn and S. Guha. Laminar families and metric embeddings: Non-bipartite maximum matching problem in the semi-streaming model. *Manuscript, available at http://arxiv.org/abs/1104.4058*, 2011.

[4] K. J. Ahn, S. Guha, and A. McGregor. Spectral sparsification in dynamic graph streams. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 1–10. Springer, 2013.

[5] A. Andoni, A. Gupta, and R. Krauthgamer. Towards (1+ ε)-approximate flow sparsifiers. In *SODA*, pages 279–293. SIAM, 2014.

[6] S. Assadi, S. Khanna, and Y. Li. Tight bounds for single-pass streaming complexity of the set cover problem. In *STOC*, pages 698–711. ACM, 2016.

[7] S. Assadi, S. Khanna, Y. Li, and G. Yaroslavtsev. Maximum matchings in dynamic graph streams and the simultaneous communication model. In *SODA*, pages 1345–1364. SIAM, 2016.

[8] A. Badanidiyuru, S. Dobzinski, H. Fu, R. Kleinberg, N. Nisan, and T. Roughgarden. Sketching valuation functions. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 1025–1035. SIAM, 2012.

[9] A. Badanidiyuru, B. Mirzasoleiman, A. Karbasi, and A. Krause. Streaming submodular maximization: Massive data summarization on the fly. In *KDD*, 2014.

[10] M. Bateni, H. Esfandiari, and V. Mirrokni. Distributed coverage maximization via sketching. *arXiv preprint arXiv:1612.02327*, 2016.

[11] G. E. Blelloch, H. V. Simhadri, and K. Tangwongsan. Parallel and I/O efficient set covering algorithms. In *SPAA*, pages 82–90, 2012.

[12] A. Borodin, H. C. Lee, and Y. Ye. Max-sum diversification, monotone submodular functions and dynamic updates. In *PODS*, pages 155–166, 2012.

[13] A. Chakrabarti and A. Wirth. Incidence geometries and the pass complexity of semi-streaming set cover. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1365–1373. SIAM, 2016.

[14] F. Chierichetti, R. Kumar, and A. Tomkins. Max-Cover in Map-Reduce. In *WWW*, pages 231–240, 2010.

[15] R. Chitnis, G. Cormode, H. Esfandiari, M. Hajiaghayi, A. McGregor, M. Monemizadeh, and S. Vorotnikova. Kernelization via sampling with applications to finding matchings and related problems in dynamic graph streams. In *SODA*, pages 1326–1344. SIAM, 2016.

[16] G. Cormode, M. Datar, P. Indyk, and S. Muthukrishnan. Comparing data streams using hamming norms (how to zero in). *Knowledge and Data Engineering, IEEE Transactions on*, 15(3):529–540, 2003.

[17] G. Cormode, H. J. Karloff, and A. Wirth. Set cover algorithms for very large datasets. In *CIKM*, pages 479–488, 2010.

[18] E. D. Demaine, P. Indyk, S. Mahabadi, and A. Vakilian. On streaming and communication complexity of the set cover problem. In *DISC*, pages 484–498. Springer, 2014.

[19] Y. Emek and A. Rosén. Semi-streaming set cover. In *Automata, Languages, and Programming*, pages 453–464. Springer, 2014.

[20] L. Epstein, A. Levin, J. Mestre, and D. Segev. Improved approximation guarantees for weighted matching in the semi-streaming model. *SIAM J. Discrete Math.*, 25(3):1251–1265, 2011.

[21] H. Esfandiari, M. T. Hajiaghayi, V. Liaghat, M. Monemizadeh, and K. Onak. Streaming algorithms for estimating the matching size in planar graphs and beyond. In *SODA*, pages 1217–1233. SIAM, 2015.

[22] U. Feige. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.

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

[24] J. Feigenbaum, S. Kannan, A. McGregor, S. Suri, and J. Zhang. On graph problems in a semi-streaming model. *Theoretical Computer Science*, 348(2):207–216, 2005.

[25] S. Har-Peled, P. Indyk, S. Mahabadi, and A. Vakilian. Towards tight bounds for the streaming set cover problem. In *PODS*, 2016.

[26] A. Hassidim and Y. Singer. Submodular optimization under noise. *arXiv preprint arXiv:1601.03095*, 2016.

[27] P. Indyk, S. Mahabadi, M. Mahdian, and V. Mirrokni. Composable core-sets for diversity and coverage maximization. In *ACM PODS*, 2014.

[28] K. Joag-Dev and F. Proschan. Negative association of random variables with applications. *The Annals of Statistics*, pages 286–295, 1983.

[29] B. Kalyanasundaram and G. Schintger. The probabilistic communication complexity of set intersection. *SIAM Journal on Discrete Mathematics*, 5(4):545–557, 1992.

[30] M. Kapralov, S. Khanna, and M. Sudan. Approximating matching size from random streams. In *SODA*, pages 734–751. SIAM, 2014.

[31] M. Kapralov, S. Khanna, and M. Sudan. Streaming lower bounds for approximating MAX-CUT. In *SODA*, pages 1263–1282. SIAM, 2015.

[32] J. A. Kelner and A. Levin. Spectral sparsification in the semi-streaming setting. In *STACS*, pages 440–451, 2011.

[33] C. Konrad, F. Magniez, and C. Mathieu. Maximum matching in semi-streaming with few passes. In *APPROX-RANDOM*, pages 231–242, 2012.

[34] C. Konrad and A. Rosén. Approximating semi-matchings in streaming and in two-party communication. In *ICALP*, pages 637–649, 2013.

[35] R. Kumar, B. Moseley, S. Vassilvitskii, and A. Vattani. Fast greedy algorithms in MapReduce and streaming. In *SPAA*, pages 1–10, 2013.

[36] A. McGregor and H. T. Vu. Better streaming algorithms for the maximum coverage problem. *arXiv preprint arXiv:1610.06199*, 2016.

[37] V. S. Mirrokni and M. Zadimoghaddam. Randomized

composable core-sets for distributed submodular maximization. In *STOC*, pages 153–162, 2015.

[38] B. Mirzasoleiman, A. Karbasi, R. Sarkar, and A. Krause. Distributed submodular maximization: Identifying representative elements in massive data. In *NIPS*, pages 2049–2057, 2013.

[39] S. Muthukrishnan. *Data streams: Algorithms and applications*. Now Publishers Inc, 2005.

[40] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functionsâĂŤi. *Mathematical Programming*, 14(1):265–294, 1978.

[41] N. Nisan. The communication complexity of approximate set packing and covering. In *Automata, Languages and Programming*, pages 868–875. Springer, 2002.

[42] A. Panconesi and A. Srinivasan. Randomized distributed edge coloring via an extension of the Chernoff-Hoeffding bounds. *SIAM Journal on Computing*, 26(2):350–368, 1997.

[43] A. A. Razborov. On the distributional complexity of disjointness. *Theoretical Computer Science*, 106(2):385–390, 1992.

[44] B. Saha and L. Getoor. On maximum coverage in the streaming model & application to multi-topic blog-watch. In *SDM*, volume 9, pages 697–708. SIAM, 2009.

APPENDIX

A. OMITTED PROOFS FOR THE SKETCHING TECHNIQUE

PROOF PROOF OF LEMMA 2.2. Let X_u be a random variable indicating whether $h(u) \leq p$ for a vertex $u \in \Gamma(G,S)$. By definition we have $\mathcal{C}(S) = |\Gamma(G,S)|$ and $\sum_{u\in\Gamma(G,S)} X_u = |\Gamma(H_p,S)|$. Thus, we have

$$\begin{aligned}\mathbf{E}\Big[|\Gamma(H_p,S)|\Big] &= \mathbf{E}\left[\sum_{u\in\Gamma(G,S)} X_u\right]\\ &= \sum_{u\in\Gamma(G,S)} \mathbf{E}[X_u]\\ &= \sum_{u\in\Gamma(G,S)} p\\ &= p|\Gamma(G,S)|.\end{aligned}$$

By the Chernoff bound to $\Gamma(H_p,S)$ we know that with probability at least $1-2\exp\left(-\frac{\varepsilon'^2 p|\Gamma(G,S)|}{3}\right)$,

$$\Big||\Gamma(H_p,S)| - p|\Gamma(G,S)|\Big| \leq \varepsilon' p|\Gamma(G,S)|.$$

In other words,

$$\mathbf{Pr}\left(\Big|\frac{1}{p}|\Gamma(H_p,S)| - \mathcal{C}(S)\Big| \leq \varepsilon'\mathcal{C}(S)\right) \geq$$
$$1-2\exp\left(-\frac{\varepsilon'^2 p\mathcal{C}(S)}{3}\right).$$

Setting $\varepsilon' = \varepsilon\frac{\mathsf{Opt}_k}{\mathcal{C}(S)}$ in the above yields

$$\begin{aligned}&\mathbf{Pr}\left(\Big|\frac{1}{p}|\Gamma(H_p,S)| - \mathcal{C}(S)\Big| \leq \varepsilon\mathsf{Opt}_k\right)\\ &\geq 1-2\exp\left(-\frac{\varepsilon^2\mathsf{Opt}_k^2 p}{3\mathcal{C}(S)}\right)\\ &\geq 1-2\exp\left(-\frac{\varepsilon^2\mathsf{Opt}_k^2}{3\mathcal{C}(S)}\frac{6\delta'}{\varepsilon^2\mathsf{Opt}_k}\right) && \text{by definition of } p,\\ &= 1-2\exp\left(-\frac{\mathsf{Opt}_k}{3\mathcal{C}(S)}6\delta'\right)\\ &\geq 1-2\exp\left(-\frac{6\delta'}{3}\right) && \text{since } \mathsf{Opt}_k \geq \mathcal{C}(S),\\ &> 1-\exp\left(1-\frac{6\delta'}{3}\right) && \text{as } 2<e,\\ &\geq 1-e^{\delta'}.\end{aligned}$$

PROOF PROOF OF LEMMA 2.5. By applying Lemma 2.3 to $S = \arg\max_S |\Gamma(H_p,S)|$, we have

$$\frac{1}{p}\max_{S\subseteq\mathcal{S}:|S|=k}|\Gamma(H_p,S)| - \mathcal{C}(S) \leq \varepsilon\mathsf{Opt}_k.$$

Combining with $\mathcal{C}(S) \leq \mathsf{Opt}_k$ and noting that H'_p is a subgraph of H_p gives

$$\max_{S\subseteq\mathcal{S}:|S|=k}|\Gamma(H'_p,S)| \leq p(1+\varepsilon)\mathsf{Opt}_k$$

and we plug in the definition of p to obtain

$$\begin{aligned}\max_{S\subseteq\mathcal{S}:|S|=k}\Gamma(H'_p,S) &\leq \frac{6C\log(n)k\delta}{\varepsilon^2\mathsf{Opt}_k}(1+\varepsilon)\mathsf{Opt}_k\\ &= \frac{6C(1+\varepsilon)\log(n)k\delta}{\varepsilon^2}\\ &\leq \frac{12C\log(n)k\delta}{\varepsilon^2}.\end{aligned}$$

PROOF PROOF OF LEMMA 2.8. Let $\mathsf{Opt}_{k'}$ be the set cover of size k' on the input graph. Pick p^* such that $H'_{p^*} = H_{\leq n}$. Remark that H_{p^*} is an induced subgraph of G containing all sets $\mathcal{S}$. Thus, $\mathsf{Opt}_{k'}$ is a set cover in H_{p^*}, as well.

Similarly to the proof of Lemma 2.4, we present here a randomized solution R^* that, in expectation, covers $1-\xi\varepsilon$ fraction of the vertices. This implies that there exists a solution of size k' covering at least $1-\xi\varepsilon$ fraction of the elements.

We construct R^* by removing $\xi\varepsilon k' = \varepsilon k$ sets chosen uniformly at random from $\mathsf{Opt}_{k'}$ and adding $\xi\varepsilon k' = \varepsilon k$ other sets picked uniformly at random.

Each element with degree at most $\frac{n\log(1/\varepsilon)}{\varepsilon k}$ in H_{p^*} still exists in $\Gamma(H_{p^*},R^*)$ with probability $1-\xi\varepsilon$, hence it is in $\Gamma(H_{\leq n},R^*)$, too. On the other hand, for any element u with degree at least $\frac{n\log(1/\varepsilon)}{\varepsilon k}$ in H_{p^*}, the probability that u is not contained by any of $\xi\varepsilon k'$ randomly chosen sets cannot exceed

$$\left(1-\frac{\frac{n\log(1/\varepsilon)}{\varepsilon k}}{n}\right)^{\varepsilon k} = \left(1-\frac{\log(1/\varepsilon)}{\varepsilon k}\right)^{\varepsilon k} \leq \left(\frac{1}{e}\right)^{\log\frac{1}{\varepsilon}} = \varepsilon.$$

Thus, each element of H_{p^*} exists in $\Gamma(H_{\leq n},R^*)$, as well, with probability $1-\varepsilon$.

B. OMITTED PROOFS FOR THE STREAMING SETTING

Proof Proof of Theorem 3.1. The first part of the statement is derived from the approximation guarantee of Greedy and Theorem 2.7. These two imply that $\mathsf{Greedy}(k, H_{\leq n}(k, \varepsilon', \delta''))$ is a $1 - \frac{1}{e} - 12\varepsilon' = 1 - \frac{1}{e} - \varepsilon$ approximate solution with probability $1 - 3e^{\delta''} = 1 - 3e^{2+\log n} \geq 1 - \frac{1}{n}$, as desired.

By definition of $H_{\leq n}(k, \varepsilon', \delta'')$, the number of edges in this sketch is not more than

$$\frac{24n\log(1/\varepsilon')\log(n)\delta''\log\log_{1-\varepsilon'} m}{(1-\varepsilon')\varepsilon'^3} + n$$
$$= \frac{24n\log(\frac{12}{\varepsilon})\log n(2+\log n)\log\log_{1-\varepsilon/12} m)}{(1-\varepsilon/12)(\varepsilon/12)^3} + n$$
$$\in O\Big(\frac{n\log^3 n\log^2 m\log\log m}{\varepsilon'^3}\Big) \in \tilde{O}(n)$$

where with out loss of generality we assume $\varepsilon' \in \Omega(\frac{1}{m})$.

Lemma B.1. *Let k' be the size of a set cover of graph G. Pick arbitrary $\varepsilon \in (0,1]$, $\lambda' \in (0, \frac{1}{e}]$, and let $k = \log(\frac{1}{\lambda'})k'$. Then $\mathsf{Greedy}(k, H_{\leq n})$ covers at least $1 - \lambda' - \varepsilon\log\frac{1}{\lambda'}$ fraction of elements $\mathcal{E}$ in $H_{\leq n}$.*

Proof. Lemma 2.8 ensures the existence of a solution of size k' on $H_{\leq n}$ covering at least $1 - \varepsilon\log\frac{1}{\lambda'}$ fraction of $\mathcal{E}$ in $H_{\leq n}$. Thus, $\mathsf{Greedy}(k, H_{\leq n})$ covers at least

$$(1-\lambda')\left[1 - \varepsilon\log\frac{1}{\lambda'}\right] \geq 1 - \lambda' - \varepsilon\log\frac{1}{\lambda'}$$

fraction of $\mathcal{E}$ in $H_{\leq n}$.

Proof Proof of Lemma 3.2. According to Lemma B.1, for a graph G with a set cover of size k', the solution $\mathsf{Greedy}(k, H_{\leq n})$ covers at least $1 - \lambda' - \varepsilon\log\frac{1}{\lambda'}$ fraction of $\mathcal{E}$ in $H_{\leq n}$, hence Algorithm 4 does not return false, as claimed.

We prove the second part of the lemma in the case Theorem 2.7 holds for $H_{\leq n}$. This happens with probability $1 - 3e^{\delta''} = 1 - 3e^{\log(C'n)+2} \geq 1 - \frac{1}{C'n}$.

On the other hand, in case Algorithm 4 does not return false, $\mathsf{Greedy}(k, H_{\leq n})$ covers no less than $1 - \lambda' - \varepsilon\log\frac{1}{\lambda'}$ fraction of $\mathcal{E}$ in $H_{\leq n}$. Then Theorem 2.7 implies that $\mathsf{Greedy}(k, H_{\leq n})$ covers at least

$$1 - \lambda' - \varepsilon\log\frac{1}{\lambda'} - 12\varepsilon \geq 1 - \lambda' - 13\varepsilon\log\frac{1}{\lambda'} = 1 - \lambda' - \varepsilon'$$

fraction of elements.

By definition of $H_{\leq n}(k, \varepsilon, \delta'')$, the number of edges in this sketch is at most

$$\frac{24n\log(1/\varepsilon)\log(n)\delta''\log\log_{1-\epsilon} m}{(1-\varepsilon)\varepsilon^3} + n$$
$$= \frac{24n\log(1/\varepsilon)\log n\log_{1+\varepsilon} n[\log(C'n)+2]}{\log\log_{1-\varepsilon} m}(1-\varepsilon)\varepsilon^3 + n$$
$$\in O\Big(\frac{n\log^2 n\log^2 m\log C'\log\log m}{\varepsilon^3}\Big) \quad \text{wlog } \varepsilon \in \Omega(\frac{1}{m}),$$
$$= O\left(\frac{n\log^2 n\log^2 m\log C'\log\log m}{\left[\frac{\varepsilon'}{13\log(1/\lambda')}\right]^3}\right)$$
$$= O\left(\frac{n\log^2 n\log^5 m\log C'\log\log m}{\varepsilon'^3}\right) \quad \text{wlog } \lambda' \in \Omega(\frac{1}{m}),$$
$$\subseteq O\left(\frac{n\log^2 n\log^6 m\log C'}{\varepsilon'^3}\right).$$

C. AN $O(NK)$ SKETCH USING ℓ_0 SKETCHES

We use the ℓ_0 sketch defined as follows.

Definition C.1 (ℓ_0 sketch)**.** *Given a multiset Π, the number of distinct elements in Π is said to be ℓ_0 of Π.*

Cormode et al. [16] provides an $O(\log n\frac{1}{\varepsilon^2}\log\frac{1}{\delta})$ space streaming algorithm that with probability $1-\delta$ gives a $1-\varepsilon$ approximation to the ℓ_0 sketch. Interestingly, one can merge two of these ℓ_0 sketches and again with probability $1-\delta$ get a $1-\varepsilon$ approximation to the ℓ_0 sketch of the merged multiset.

Given an input graph $G(\mathcal{S}, \mathcal{E})$, we maintain an ℓ_0 sketch (using the algorithm in [16]) for the set of neighbors of each vertex in $\mathcal{S}$ (we fix ε and δ used in the ℓ_0 sketch later). We estimate the value of each set $S \subseteq \mathcal{S}$ by merging the ℓ_0 sketches corresponding to the vertices in S.

Consider that, for each set $S \subseteq \mathcal{S}$, with probability $1-\delta$, we give a $1-\varepsilon$ approximation of the coverage valuation of S. Nevertheless, to find the k-cover solution, we look at $\binom{n}{k}$ sets. The union bound ensures that, with probability $1 - \binom{n}{k}\delta$, all the $\binom{n}{k}$ estimations are accurate. We set $\delta = \frac{1}{\tilde{\Theta}(\binom{n}{k})}$, so these hold with high probability.

The space required by each ℓ_0 sketch is $O(\log n\frac{1}{\varepsilon^2}\log\frac{1}{\delta}) = \tilde{O}(\log\binom{n}{k}) = \tilde{O}(k)$. The space required by the algorithm to maintain n sketches is thus $\tilde{O}(nk)$, giving the following theorem.

Theorem C.2. *Using ℓ_0 sketches, there exists an exponential-time $1-\varepsilon$ approximation streaming algorithm for k cover using $\tilde{O}(nk)$ space.*