Semi-Streaming Set Cover

Yuval Emek, Technion - Israel Institute of Technology
Adi Rosén, CNRS and Université Paris Diderot, France

This paper studies the set cover problem under the semi-streaming model. The underlying set system is formalized in terms of a hypergraph $G = (V, E)$ whose edges arrive one-by-one and the goal is to construct an edge cover $F \subseteq E$ with the objective of minimizing the cardinality (or cost in the weighted case) of F. We further consider a parameterized relaxation of this problem, where given some $0 \leq \epsilon < 1$, the goal is to construct an edge $(1-\epsilon)$-cover, namely, a subset of edges incident to all but an ϵ-fraction of the vertices (or their benefit in the weighted case). The key limitation imposed on the algorithm is that its space is limited to (poly)logarithmically many bits per vertex.

Our main result is an asymptotically tight trade-off between ϵ and the approximation ratio: We design a semi-streaming algorithm that on input hypergraph G, constructs a succinct data structure $\mathcal{D}$ such that for every $0 \leq \epsilon < 1$, an edge $(1-\epsilon)$-cover that approximates the optimal edge (1-)cover within a factor of $f(\epsilon, n)$ can be extracted from $\mathcal{D}$ (efficiently and with no additional space requirements), where

$$f(\epsilon, n) = \begin{cases} O(1/\epsilon), & \text{if } \epsilon > 1/\sqrt{n} \\ O(\sqrt{n}), & \text{otherwise} \end{cases} .$$

In particular for the traditional set cover problem we obtain an $O(\sqrt{n})$-approximation. This algorithm is proved to be best possible by establishing a family (parameterized by ϵ) of matching lower bounds.

Categories and Subject Descriptors: F.2.0 [**Analysis of Algorithms and Complexity**]: General

General Terms: Algorithms, Theory

Additional Key Words and Phrases: streaming algorithms, set cover, lower bounds

ACM Reference Format:
Yuval Emek and Adi Rosén, 2016. Semi-Streaming Set Cover. *ACM* V, N, Article A (August 2016), 23 pages.
DOI: 1.1

1. INTRODUCTION

Given a *set system* consisting of a *universe* of items and a collection of item sets, the goal in the *set cover* problem is to construct a minimum cardinality subcollection of sets that covers the whole universe. This problem is fundamental to combinatorial optimization with applications ranging across many different domains. It is one of the 21 problems whose NP-hardness was established by Karp [Karp 1972] and its study has led to the development of various techniques in the field of approximation algorithms (see, e.g., [Vazirani 2001]).

In this paper, we investigate the set cover problem under the *semi-streaming* model [Feigenbaum et al. 2005], where the sets arrive one-by-one and the algorithm's space is constrained to maintaining a small number of bits per item (cf. the *set-streaming* model [Saha and Getoor 2009]). In particular, we are interested in the following two questions: (1) What is the best approximation ratio for the set cover problem under

A preliminary version of this paper appeared in the proceedings of ICALP 2014 [Emek and Rosén 2014].
Research of the second author partially supported by ANR project RDAM.
Author's addresses: Yuval Emek, Faculty of Industrial Engineering and Management, Technion - Israel Institute of Technology, Technion City, Haifa 3200003, Israel, yemek@ie.technion.ac.il; Adi Rosén, IRIF - Case 7014, Université Paris Diderot, 75205 Paris Cedex 13, France, adiro@liafa.univ-paris-diderot.fr.

such memory constraints? (2) How does the answer to (1) change if we relax the set cover notion so that the set subcollection is required to cover only a δ-fraction of the universe?

On top of the theoretical interest in the aforementioned questions, studying the set cover problem under the semi-streaming model is also justified by several practical applications. For example, Saha and Getoor [Saha and Getoor 2009] describe the setting of a web crawler that iterates a large collection of blogs, listing the topics covered by each one of them. A user interested in a certain set of topics can run a semi-streaming set cover algorithm with relatively small memory requirements to identify a subcollection of blogs that covers the desired topics.

The model. In order to fit our terminology to the graph theoretic terminology traditionally used in the semi-streaming literature (and also to ease the presentation), we use an equivalent formulation for the set cover problem in terms of edge covers in hypergraphs: Consider some *hypergraph* $G = (V, E)$, where V is a set of n *vertices* and E is a (multi-)set of m *hyperedges* (henceforth *edges*), where each edge $e \in E$ is an arbitrary non-empty subset $e \subseteq V$. Assume hereafter that G does not admit any isolated vertices, namely, every vertex is incident to at least one edge. We say that an edge subset $F \subseteq E$ *covers* G if every vertex in V is incident to some edge in F. The goal of the *edge cover* problem is to construct a subset $F \subseteq E$ of edges that covers G, where the objective is to minimize the cardinality $|F|$.

A natural relaxation of the covering notion seeks to cover some fraction of the vertices in V: Given some $0 < \delta \leq 1$, we say that an edge subset $F \subseteq E$ δ-*covers* G if at least δn vertices are incident to the edges in F, namely, $|V(F)| \geq \delta n$, where $V(F) = \{v \in V \mid \exists e \in F \text{ s.t. } v \in e\}$. Under this terminology, a cover of G is referred to as a 1-cover. This raises a bi-criteria optimization version of the set cover problem, where the goal is to construct an edge subset $F \subseteq E$ that δ-covers G with the objective of minimizing $|F|$ and maximizing δ. In this paper, we focus on approximation algorithms, where the cardinality of F is compared to that of an optimal edge (1-)cover of G.

In the *weighted* version of the edge cover problem, the hypergraph G is augmented with vertex *benefits* $b : V \to \mathbb{Q}_{>0}$ and edge *costs* $c : E \to \mathbb{Q}_{>0}$. The edge cover definition is generalized so that edge subset $F \subseteq E$ is said to δ-*cover* G if the benefit of the vertices incident to the edges in F is at least a δ-fraction of the total benefit, namely, $b(V(F)) \geq \delta \cdot b(V)$, where $b(U) = \sum_{v \in U} b(v)$ for every vertex subset $U \subseteq V$. The goal is then to construct an edge subset F that δ-covers $G = (V, E, b, c)$, where the objective is to maximize δ and minimize the cost of F, denoted $c(F) = \sum_{e \in F} c(e)$.

Under the *semi-streaming* model, the execution of an algorithm is partitioned into discrete time steps and the edges in E are presented one-by-one so that edge $e_t \in E$ is presented at time $t = 0, 1, \ldots, m-1$, listing all vertices $v \in e_t$;[1] in the weighted version, the cost of e_t and the benefits of the vertices it contains are also listed. The key limitation imposed on the algorithm is that its space is limited; specifically, we allow the algorithm to maintain $\log^{O(1)} |G|$ bits per vertex, where $|G|$ denotes the number of bits in the standard binary encoding of G. Each edge $e \in E$ is associated with a unique *identifier* $\mathrm{id}(e)$ of size $O(\log m)$ bits, say, the time t at which edge e_t is presented. We may sometimes use the identifier $\mathrm{id}(e)$ when we actually refer to the edge e itself, e.g., replacing $c(e)$ with $c(\mathrm{id}(e))$; our intention will be clear from the context.

In contrast to the random access memory model of computation, where, given a collection $\mathcal{I}$ of identifiers, one can verify that all vertices in V are covered by the the edges

[1]With the exception of our related work discussion, all semi-streaming algorithms in this paper make a single (one way) pass over the input hypergraph.

whose identifiers are in $\mathcal{I}$ (and also determine which vertex in V is incident to which of the edges), under the semi-streaming model, the collection $\mathcal{I}$ by itself typically fails to provide this information. Therefore, instead of merely returning the identifiers of some edge δ-cover, we require that the algorithm's output encodes the collection $\mathcal{I}$ of edge identifiers together with, for each vertex v, an identifier in $\mathcal{I}$ of an edge incident to v (or null if v is not covered by the edge δ-cover). Formally, the algorithm is required to output a *δ-cover certificate* χ for G which is a partial function from V to $\{\mathrm{id}(e) \mid e \in E\}$ with *domain*

$$\mathrm{Dom}(\chi) = \{v \in V \mid \chi \text{ is defined over } v\}$$

and *image*

$$\mathrm{Im}(\chi) = \{\mathrm{id}(e) \mid \exists v \in \mathrm{Dom}(\chi) \text{ s.t. } \chi(v) = \mathrm{id}(e)\}$$

that satisfies (1) if $v \in \mathrm{Dom}(\chi)$ and $\chi(v) = \mathrm{id}(e)$, then $v \in e$; and (2) $b(\mathrm{Dom}(\chi)) \geq \delta \cdot b(V)$. By definition, the image of χ consists of the identifiers of the edges in some edge δ-cover F of G and the quality of the δ-cover certificate χ is thus measured in terms of $c(\mathrm{Im}(\chi)) = c(F)$.

Our contributions. Consider some unweighted hypergraph $G = (V, E)$ with optimal edge 1-cover $\mathtt{OPT}$. We design a deterministic semi-streaming algorithm, referred to as $\mathtt{SSSC}$ (acronym of the paper's title), for the edge (δ-)cover problem that, given some $0 \leq \epsilon < 1$, outputs a $(1-\epsilon)$-cover certificate χ_ϵ for G with image of cardinality $|\mathrm{Im}(\chi_\epsilon)| = O(\min\{1/\epsilon, \sqrt{n}\} \cdot |\mathtt{OPT}|)$.[2] This result is extended to the weighted case, where $G = (V, E, b, c)$, showing that $c(\mathrm{Im}(\chi_\epsilon)) = O(\min\{1/\epsilon, \sqrt{n}\} \cdot c(\mathtt{OPT}))$ (see Thm. 2.2 and Thm. 2.3). In particular, for the edge (1-)cover problem, we obtain an $O(\sqrt{n})$-approximation for both the weighted and unweighted cases.

On the negative side, we prove that for every $\epsilon \geq 1/\sqrt{n}$, if a randomized semi-streaming algorithm for the set cover problem outputs a $(1-\epsilon)$-cover certificate χ for G, then it cannot guarantee that $\mathbb{E}[|\mathrm{Im}(\chi)|] = o(|\mathtt{OPT}|/\epsilon)$ (see Thm. 3.1). This demonstrates that the approximation guarantee of our algorithm is asymptotically optimal for the whole range of parameter $0 \leq \epsilon < 1$ even for randomized algorithms.

We note that $\mathtt{SSSC}$ has the attractive feature that the (near-linear size) data structure that it maintains is oblivious to the parameter ϵ. That is, the algorithm processes the stream of edges with no knowledge of ϵ, generating a data structure $\mathcal{D}$, and the promised $(1-\epsilon)$-cover certificate χ_ϵ can be efficiently extracted from $\mathcal{D}$ (with no additional space requirements) for every $0 \leq \epsilon < 1$ (in fact several such covers for different values of ϵ can be extracted). From a bi-criteria optimization perspective, our lower bound implies that the parameterized collection $\{\chi_\epsilon\}_{0 \leq \epsilon < 1}$ encoded in $\mathcal{D}$ is an (asymptotically) optimal solution frontier (cf. Pareto optimality).

Using a simple adjustment of the randomized rounding technique for set cover (see, e.g., [Vazirani 2001]), it is not difficult to show that a basic feasible solution to the linear program relaxation $\mathcal{P}$ of a given set cover instance also serves as a compact data structure from which a $(1-\epsilon)$-cover certificate χ_ϵ can be extracted for every $0 \leq \epsilon < 1$. In fact, the approximation ratio obtained this way is better than ours, namely, $O(\log(1/\epsilon))$. However, our lower bound shows that this approach cannot be applied — and in passing, that the linear program relaxation $\mathcal{P}$ cannot be solved — under the semi-streaming model.

Can our tight lower bound be an artifact of the requirement that the algorithm outputs a cover *certificate*? We nearly eliminate this possibility by proving that for every constant $c > 0$ and for every $\epsilon \geq n^{-1/2+c}$, even if the randomized algorithm has to

[2]Define $\min\{1/x, y\} = y$ when $x = 0$.

return only an "uncertified" output, i.e., only the identifiers of the edges in some edge $(1-\epsilon)$-cover F of G are returned, then the expected cardinality of F must still be large, specifically, $\mathbb{E}[|F|] = \Omega\left(\frac{\log \log n}{\log n} \cdot |\mathtt{OPT}|/\epsilon\right)$, where OPT in this case is proportional to $\epsilon^2 n$ (see Thm. 3.2).[3]

Related work. The work most closely related to the present paper is probably the one presented in Saha and Getoor's paper [Saha and Getoor 2009] that also considers the set cover problem under a variant of the streaming model called *set-streaming*, formulated as the edge cover problem in hypergraphs. Saha and Getoor design a 4-approximation algorithm for the *maximum coverage* problem that, given a hypergraph $G = (V, E)$ and a parameter k, looks for k edges that cover as many vertices as possible, where the algorithm stores (explicitly) at most k edges at any given time. Based on that, they observe that an $O(\log n)$-approximation for the optimal set cover can be obtained in $O(\log n)$ passes over the input (this can be achieved based on our semi-streaming algorithm as well). Using the terminology of the present paper, Saha and Getoor's maximum coverage algorithm is very efficient for obtaining edge $(1-\epsilon)$-covers as long as ϵ is large, but it does not provide any (single pass) guarantees for $\epsilon < 3/4$. In contrast, our algorithm has asymptotically optimal (single pass) guarantees for any $0 \leq \epsilon < 1$.

Subsequent to the initial publication of the present work [Emek and Rosén 2014], a series of papers considered the set cover problem in the streaming setting, under the semi-streaming space constraint (i.e., space in $O(n \text{ polylog}(n, m))$), or the sublinear space constraint (i.e., space in $o(mn)$).

Assasi, Khanna, and Li [Assadi et al. 2016] generalize our upper and lower bounds for 1-cover and show that in order to achieve α-approximation ($\alpha = o(\sqrt{n})$), $\tilde{\Theta}(mn/\alpha)$ space is necessary and sufficient for deterministic and randomized one-pass streaming algorithms. If only an estimation of the size of the optimal set cover is sought (rather than finding the cover itself), then $\tilde{\Theta}(mn/\alpha^2)$ space is necessary and sufficient for randomized one-pass streaming algorithms.

Chakrabarti and Wirth [Chakrabarti and Wirth 2016] give essentially tight bounds on the approximation ratio achievable by deterministic and randomized semi-streaming algorithms that use $p \geq 1$ passes. They show that the smallest α for which a (deterministic or randomized) semi-streaming algorithm can compute an α-approximation $(1-\epsilon)$-cover for unweighted instances is $\Theta\left(\min\left\{n^{\frac{1}{p+1}}, \epsilon^{-\frac{1}{p}}\right\}\right)$ (ignoring multiplicative factors involving p). Their results imply for our one-pass setting a somewhat better lower bound than ours in that they show a lower bound on the approximation ratio of algorithms that only have to approximate the size of the minimum (partial) set cover rather than finding one.

Demaine et al. [Demaine et al. 2014] study sublinear-space streaming algorithms and consider the interplay between the number of passes and the approximation ratio as well as the space requirements. They show that for any $\delta = \Omega(1/\log n)$, an approximation ratio of $O(4^{\frac{1}{\delta}})$ can be achieved in $O(4^{\frac{1}{\delta}})$ passes using $\tilde{O}(mn^{\delta})$ space, if expo-

[3]By using a rather simple reduction from the index function studied in communication complexity [Kremer et al. 1999], one can show that there does not exist a semi-streaming algorithm that distinguishes between hypergraphs admitting a constant size edge cover and hypergraphs that cannot be covered by less than n^{α} edges for any constant $0 < \alpha < 1/2$. This lower bound is more attractive in the sense that it applies already to the decision version of the set cover problem. However, to the best of our understanding, in contrast to the constructions of the present paper, this simple reduction cannot be generalized to $(1-\epsilon)$-covers for values of $\epsilon \gg 1/\sqrt{n}$. We note that subsequent to the initial publication of the present work [Emek and Rosén 2014], Chakrabarti and Wirth [Chakrabarti and Wirth 2016] obtained such a generalization using more complicated combinatorial structures than those used for the simpler reduction.

nential time is allowed. A multiplicative factor of ρ is added to the space and approximation bounds, if the run-time is polynomial and a ρ-approximation standard off-line algorithm for set cover is used as a subroutine (rather than using as a subroutine an exponential time optimal algorithm for set cover). They further give lower bounds for deterministic streaming algorithms showing that whatever the number of passes is, with space in $o(mn)$, one cannot achieve a constant approximation ratio. These results were later improved by Indyk, Mahabadi, and Vakilian [Indyk et al. 2015], who gave an $O(1/\delta)$-pass randomized algorithm that achieves an $O(\rho/\delta)$-approximation ratio using $\tilde{O}(mn^{\delta})$ space.

The semi-streaming model was introduced by Feigenbaum et al. [Feigenbaum et al. 2005] for graph theoretic problems, where the edges of an n vertex input graph arrive sequentially and the algorithm is allowed to maintain only $\log^{O(1)} n$ bits of memory per vertex. Since the number of bits required to encode an n vertex graph is $n^{O(1)}$, the space-per-vertex bound used in the present paper can be viewed as a generalization of the bound used by Feigenbaum et al. from graphs to hypergraphs. In any case, if one restricts attention to hypergraphs with $m \leq 2^{\log^{O(1)} n}$ edges then the two bounds are identical (refer to Sec. 2 for further discussions on the space bounds of our algorithm).

A large number of graph theoretic problems have been treated under the streaming model in recent years, cf. [McGregor 2014]. For example, matching problems (e.g., [McGregor 2005; Epstein et al. 2011; Konrad et al. 2012; Esfandiari et al. 2015]), connectivity and minimum spanning tree problems (e.g., [Feigenbaum et al. 2005]), diameter and shortest path problems (e.g., [Feigenbaum et al. 2008]), min-cut, max-cut, and sparsification problems (e.g., [Ahn and Guha 2009; Kelner and Levin 2013; Kapralov et al. 2014; Kapralov et al. 2015]), problems related to the construction of graph spanners (e.g., [Baswana 2008; Elkin 2011]), and the maximum independent set problem (e.g., [Halldórsson et al. 2010; Emek et al. 2012]), to name a few.

Several variants of the set cover problem have been investigated under the model of online computation. Alon et al. [Alon et al. 2009] focus on the online problem in which some master set system is known in advance and an unknown subset of its items arrive online; the goal is to cover the arriving items, minimizing the number of sets used for that purpose. Another online variant of the set cover problem is studied by Fraigniaud et al. [Fraigniaud et al. 2016], where the sets arrive online, but not all items have to be covered. Here, each item is associated with a penalty and the cost incurred by the algorithm is the sum of the total cost of the sets chosen for the partial cover plus the total penalty of the non-covered items.

The set cover version studied in the present paper can also be considered under the online computation model with the requirement that the algorithm maintains a cover for the items seen so far. This is meaningful only if preemption is allowed (i.e., a set can be added to the cover only at the time of its arrival, but can be excluded from the cover at all times afterwards) and under a slightly stronger definition for the competitive ratio: The performance of the algorithm is measured via the maximum, over time t, of the ratio $\texttt{ALG}_t/\texttt{OPT}_t$, where $\texttt{OPT}_t$ is the cost of an optimal set cover for the set system presented up to time t, and $\texttt{ALG}_t$ is the cost of the set cover maintained by the algorithm for that set system at time t. The set cover algorithm presented in the present paper is, in fact, also an online algorithm for this problem with competitive ratio $O(\sqrt{n})$. The lower bound(s) established in the present paper can be slightly modified to show that this is optimal.

Closely related to our notion of cover certificate is the *universal set cover* problem [Jia et al. 2005; Grandoni et al. 2013], where, given a set system, the goal is to construct a mapping f from the items to the sets containing them so that for every subset of items, X, the cost of the image of X under f is as close as possible to the cost of a minimum set cover for X. This problem resembles our guarantee that the promised $(1-\epsilon)$-cover certificate can be extracted from the data structure for every ϵ. However, it is much stronger in the sense that it guarantees a small cover for every item subset, rather than the existence of a "good" item subset for every ϵ. To the best of our knowledge, the universal set cover problem has not been studied under the semi-streaming model.

Techniques' overview.

Streaming algorithm. Consider some hypergraph $G = (V, E, b, c)$ and assume for simplicity that the vertex benefits are uniform, i.e., $b(v) = 1$ for every vertex $v \in V$. Translated to the terminology of edge covers in hypergraphs, the classic greedy approximation algorithm for minimum set cover [Johnson 1974] (see also [Vazirani 2001]) associates a *price* variable with each vertex $v \in V$. This variable is set upon the first time v is covered by some picked edge $e \in E$ and it captures the cost of covering v by e, that is, $c(e)$ divided by the number of newly covered vertices. The crux of this classic algorithm is that it picks the edges so that the vertices are covered in non-decreasing order of prices.

Under the (semi-)streaming model, we have no control over the arrival order of the edges and, thus, we cannot hope to mimic the greedy algorithm. Instead, we maintain for each vertex $v \in V$, an *effectiveness* variable $\mathrm{eff}(v)$ and a variable $\mathrm{eid}(v)$. The variable $\mathrm{eid}(v)$ stores the identifier $\mathrm{id}(e_t)$ of the edge e_t that is (currently) intended to cover v. The effectiveness variable is analogous to the price variable of the greedy algorithm in the sense that it captures the quality of e_t in covering v (unlike the price variables, higher values of $\mathrm{eff}(v)$ indicate better quality, but the logic is similar). Another difference is that in our algorithm a vertex v, after first being covered, may later change the edge that covers it, and thus also its price. More formally, if $T \subseteq e_t$ is the subset of vertices v for which the algorithm assigned $\mathrm{eid}(v) \leftarrow \mathrm{id}(e_t)$ at time t, then the effectiveness variables are set to be $\mathrm{eff}(v) = \left\lceil \log \frac{|T|}{c(e_t)} \right\rceil$.[4] The key idea behind our streaming algorithm is that T is taken to be the largest subset $T \subseteq e_t$ such that the effectiveness variable $\mathrm{eff}(v)$ of every vertex $v \in T$ strictly increases at time t.

A careful analysis shows that upon termination of the input stream, there exists some threshold ρ such that the total benefit of vertices $v \in V$ with $\mathrm{eff}(v) \leq \rho$ is at most ϵn, whereas the total cost of the edges whose identifiers are stored in the $\mathrm{eid}(v)$ variables of vertices v with $\mathrm{eff}(v) > \rho$ is $O(c(\mathtt{OPT})/\epsilon)$. This provides the desired approximation for $\epsilon > 1/\sqrt{n}$.

If $\epsilon \leq 1/\sqrt{n}$, then our algorithm leaves at most $\sqrt{n}$ uncovered vertices and we cover them using the cheapest possible edge for each vertex individually. The promised approximation ratio is obtained since the total cost of these extra edges is at most $\sqrt{n} \cdot c(\mathtt{OPT})$.

Lower bounds. The hard hypergraphs that lie at the heart of our lower bound are constructed based on an *affine plane* $\mathcal{A} = (P, L)$ with q^2 points and $q(q+1)$ lines (see, e.g., [Lindner and Rodger 2011]). By randomly partitioning each line in L into two edges (or more edges in the "uncertified" version of the lower bound), we obtain the probability distribution over hypergraphs that lies at the heart of the lower bound proofs. The hard input sequence for the lower bound proof starts by presenting, for each line in L, the two edges that correspond to that line. Then the adversary presents one additional edge e^* that contains the points of all but $r \approx \epsilon q$ random lines, where all those lines belong to some single random angle A_i of $\mathcal{A}$ (refer to Figure 1 for an illustration). An optimal edge cover for such instance consists of the edge e^* and the $2r = O(\epsilon q)$

[4]Throughout, $\log$ denotes logarithm to the base of 2.

edges corresponding to the r lines missing from e^*. Using careful information theoretic arguments, we show that any low-space deterministic algorithm must use, on expectation, many lines from angles other than A_i in order to construct a $(1-\epsilon)$-cover F. The properties of affine planes guarantee that the expected cardinality of F is $\Omega(q)$. By Yao's principle, this argument is translated to a lower bound for randomized low-space algorithms.

2. A SEMI-STREAMING ALGORITHM

Our goal in this section is to design a semi-streaming algorithm for the edge (δ-)cover problem in hypergraphs. The algorithm, referred to as `SSSC` (acronym of the paper's title), is presented in Sec. 2.1 and its approximation ratio is analyzed in Sec. 2.2. For the sake of simplicity, we first assume that all numerical values (vertex benefits and edge costs) are encoded using $O(\log n)$ bits. Under this assumption, the space bounds of `SSSC` are quite trivial and together with the analysis in Sec. 2.2 yield Theorem 2.1.

THEOREM 2.1. *On a weighted input hypergraph $G = (V, E, b, c)$ with numerical values encoded using $O(\log n)$ bits, our algorithm uses $O(n \log(n+m))$ space, processes each input edge $e_t \in E$ in $O(|e_t| \log |e_t|)$ time, and produces a data structure $\mathcal{D}$ with the following guarantee: For every $0 \le \epsilon < 1$, a $(1-\epsilon)$-cover certificate χ_ϵ for G such that*

$$c(\mathrm{Im}(\chi_\epsilon)) = O\left(\min\left\{1/\epsilon, \sqrt{n}\right\} \cdot c(\mathtt{OPT})\right)$$

can be extracted from $\mathcal{D}$ in time $O(n \log n)$ with no additional space requirements, where `OPT` *stands for an optimal edge (1-)cover of G.*

Sec. 2.3 is dedicated to lifting the assumption on the numerical values. The following definitions are necessary for the discussion of the results we obtain without this assumption:

$$b^{\log} = \left\lceil \log \max_{v \in V} \left\{ b(v), b(v)^{-1} \right\} \right\rceil \qquad c^{\log} = \left\lceil \log \max_{e \in E} \left\{ c(e), c(e)^{-1} \right\} \right\rceil \qquad c^{\Delta} = \left\lceil \log \frac{\max_{e \in E} c(e)}{\min_{e \in E} c(e)} \right\rceil ,$$

where the last parameter captures the number of bits required to encode the edge costs *aspect ratio*. Note that the encoding size $|G|$ of the input weighted hypergraph $G = (V, E, b, c)$ is of size at least $b^{\log} + c^{\log}$. Moreover, c^{Δ} is always at most $2c^{\log}$, but it may be much smaller than that.

Our results are cast in Thm. 2.2 and in Thm. 2.3, where the former generalizes Thm. 2.1 and the latter has a better space bound, but a slightly worse run-time guarantee. Another drawback of Thm. 2.3 is that it requires that the parameters n and ϵ are known to the algorithm in advance in contrast to Thm. 2.1 and Thm. 2.2 that do not require an a priori knowledge of any global parameter.

THEOREM 2.2. *On a weighted input hypergraph $G = (V, E, b, c)$, our algorithm uses $O\left(n \log\left(n + m + b^{\log} + c^{\log}\right)\right)$ space, processes each input edge $e_t \in E$ in $O(|e_t| \log |e_t|)$ time, and produces a data structure $\mathcal{D}$ with the following guarantee: For every $0 \le \epsilon < 1$, a $(1-\epsilon)$-cover certificate χ_ϵ for G such that*

$$c(\mathrm{Im}(\chi_\epsilon)) = O\left(\min\left\{1/\epsilon, \sqrt{n}\right\} \cdot c(\mathtt{OPT})\right)$$

can be extracted from $\mathcal{D}$ in time $O(n \log n)$ with no additional space requirements, where `OPT` *stands for an optimal edge (1-)cover of G.*

THEOREM 2.3. *On a weighted input hypergraph $G = (V, E, b, c)$, for any $0 \le \epsilon < 1$, our algorithm (knowing n and ϵ in advance) uses $O\left(\log\left(b^{\log} + c^{\log}\right) + n \log\left(n + m + c^{\Delta}\right)\right)$ space, processes each input edge $e_t \in E$ in $O(n \log n)$ time, and outputs a $(1-\epsilon)$-cover* *certificate χ_ϵ for G such that*

$$c(\mathrm{Im}(\chi_\epsilon)) = O\left(\min\left\{1/\epsilon, \sqrt{n}\right\} \cdot c(\texttt{OPT})\right) ,$$

where OPT *stands for an optimal edge (1-)cover of G.*

2.1. The Algorithm

In what follows we consider some weighted hypergraph $G = (V, E, b, c)$ with optimal edge (1-)cover OPT. The main building block of algorithm SSSC is a procedure referred to as COVER. This procedure processes the stream of edges and outputs for every node $v \in V$, an identifier of an edge e that covers it, together with an integer variable that intuitively captures the quality of edge e in covering v. Algorithm SSSC uses two parallel invocations of COVER, one on the input graph G and one on some modification of G, and upon termination of the input stream, extracts the desired cover certificate from the output of these two invocations.

2.1.1. Procedure COVER. The procedure maintains for each vertex $v \in V$, the following variables:

— $\mathrm{eid}(v)$ = an identifier $\mathrm{id}(e)$ of some edge $e \in E$; and
— $\mathrm{eff}(v)$ = a (not necessarily positive) integer refereed to as the *effectiveness* of v.

We denote by $\mathrm{eid}_t(v)$ and $\mathrm{eff}_t(v)$ the values of $\mathrm{eid}(v)$ and $\mathrm{eff}(v)$, respectively, at time t (i.e., just before e_t is processed). Procedure COVER that relies on the following definition is presented in Algorithm 1.

Definition 2.4 (***level, effectiveness***). Consider edge e_t presented at time t and some subset $T \subseteq e_t$. The *level* of T at time t, denoted $\mathrm{lev}_t(T)$, is defined as

$$\mathrm{lev}_t(T) = \left\lceil \log \frac{b(T)}{c(e_t)} \right\rceil .$$

Subset T is said to be *effective* at time t if for every $v \in T$, it holds that

$$\mathrm{lev}_t(T) > \mathrm{eff}_t(v) .$$

Note that $\emptyset$ is always vacuously effective.

Algorithm 1 COVER$(G = (V, E, b, c))$

```
**Initialization** $\forall v \in V$: $\mathrm{eid}(v) \leftarrow \bot$ and $\mathrm{eff}(v) \leftarrow -\infty$
**for** $t = 0, 1, \ldots$ **do**
  Read edge $e_t \in E$ from the stream
  Compute an effective subset $T \subseteq e_t$ of largest benefit $b(T)$
  **for all** $v \in T$ **do**
    $\mathrm{eid}(v) \leftarrow \mathrm{id}(e_t)$
    $\mathrm{eff}(v) \leftarrow \mathrm{lev}_t(T)$
  **end for**
**end for**
**return** $\mathrm{eid}(\cdot)$ and $\mathrm{eff}(\cdot)$
```

2.1.2. Algorithm `SSSC`. We are now ready to present our algorithm `SSSC`. The first part of the algorithm is composed of four procedures referred to as P1, P2, P3, and P4, executed in parallel on the the input stream, in order to produce a certain data structure. Then, given $0 \leq \epsilon < 1$, the algorithm can extract from that data structure the desired $(1-\epsilon)$-cover.

The first procedure (P1) is Procedure `COVER`, executed on the original stream. The result of this procedure is used to extract the desired cover when $\epsilon \geq 1/\sqrt{n}$. The second procedure (P2) is Procedure `COVER`, executed on the input stream, but assuming that all vertex benefits are uniform (as if our aim is to maximize the number of covered vertices rather than their total weight). The result of this procedure is used to extract the desired cover when $\epsilon < 1/\sqrt{n}$. The last two procedures are quite trivial and they are needed only for technical purposes: procedure P3 stores the identifier of the minimum-cost set that contains vertex v for every $v \in V$; and procedure P4 stores the benefit $b(v)$ of vertex v for every $v \in V$. We note that if ϵ is known in advance, then one may run only one of the first two procedures. Nevertheless, running all four procedures in parallel allows us to go over the stream only once, and then extract a number of different covers for different values of ϵ. We now formally define the algorithm.

On input weighted graph $G = (V, E, b, c)$, algorithm `SSSC` runs in parallel the following procedures that process the stream of edges:

P1: $(\mathrm{eid}_\infty(\cdot), \mathrm{eff}_\infty(\cdot)) \leftarrow$ `COVER`$(G = (V, E, b, c))$.
P2: $(\mathrm{eid}^{\mathbf{1}}_\infty(\cdot), \mathrm{eff}^{\mathbf{1}}_\infty(\cdot)) \leftarrow$ `COVER`$(G = (V, E, \mathbf{1}, c))$, where $\mathbf{1}$ stands for the function that assigns a unit benefit to all vertices $v \in V$.
P3: A procedure that maintains for every vertex $v \in V$, a variable $\mathrm{emin}(v)$ that stores the identifier of the minimum cost edge that covers v, seen so far.
P4: A procedure that stores for every vertex $v \in V$, its benefit $b(v)$.

Upon termination of the input stream, `SSSC` takes some parameter $0 \leq \epsilon < 1$ and extracts the desired $(1-\epsilon)$-cover certificate for G from the variables returned by procedures P1–P4. We distinguish between the following two cases.

— Case $\epsilon \geq 1/\sqrt{n}$:
The algorithm looks for the largest integer r^* such that $b(I(\leq r^*)) \leq \epsilon b(V)$, where

$$I(\leq r^*) = \{v \in V : \mathrm{eff}_\infty(v) \leq r^*\},$$

and returns the partial function $\chi : V \to \mathrm{id}(E)$ that maps every vertex $v \in V - I(\leq r^*)$ to $\mathrm{eid}_\infty(v)$.
— Case $\epsilon < 1/\sqrt{n}$:
The algorithm looks for the largest integer r^* such that $|I^{\mathbf{1}}(\leq r^*)| \leq \sqrt{n}$, where

$$I^{\mathbf{1}}(\leq r^*) = \{v \in V : \mathrm{eff}^{\mathbf{1}}_\infty(v) \leq r^*\},$$

and sets χ' to be the partial function $\chi' : V \to \mathrm{id}(E)$ that maps every vertex $v \in V - I^{\mathbf{1}}(\leq r^*)$ to $\mathrm{eid}^{\mathbf{1}}_\infty(v)$. Then, it returns the (complete) function $\chi'' : V \to \mathrm{id}(E)$ extended from χ' by mapping every vertex $v \in I^{\mathbf{1}}(\leq r^*)$ to $\mathrm{emin}(v)$.

Notice that the unweighted case is much simpler: If $G = (V, E)$, then procedure P2 is identical to procedure P1; moreover, procedures P3 and P4 are redundant since all vertices/edges admit a unit benefit/cost. Further note that procedures P1–P4 are oblivious to ϵ. Upon termination of the input stream, the algorithm extracts, for the given $0 \leq \epsilon < 1$, the desired $(1-\epsilon)$-cover certificate for G from the variables returned by procedures P1–P4. In fact, several such cover certificates can be extracted for different values of ϵ.

2.2. Analysis

We begin our analysis with some observations regarding our main procedure COVER.

OBSERVATION 2.5. *If $T \subseteq e_t$ is effective at time t and $v \in T$, then $T \cup \{u\}$ is effective at time t for every $u \in e_t$ such that $\text{eff}_t(u) \leq \text{eff}_t(v)$.*

Notice that COVER's updating rule guarantees that the effectiveness $\text{eff}(v)$ is non-decreasing throughout the course of the execution. Employing Obs. 2.5, we can now derive Obs. 2.6 and Obs. 2.7 (the former follows by sorting the vertices $v \in e_t$ in non-decreasing order of the value of the effectiveness $\text{eff}(v)$ and looking for the largest effective prefix).

OBSERVATION 2.6. *The run-time of COVER on edge e_t is $O(|e_t| \log |e_t|)$.*

OBSERVATION 2.7. *If $T \subseteq e_t$ is effective at time t, then for every $v \in T$, it holds that*

$$\text{eff}_{t+1}(v) \geq \text{lev}_t(T) .$$

We are now ready to establish the following lemma.

LEMMA 2.8. *Consider some integer r. Procedure COVER guarantees that*

$$b\left(\{v \in e_t \mid \text{eff}_{t+1}(v) \leq r\}\right) < 2^{r+1} \cdot c(e_t) .$$

PROOF. Assume towards a contradiction that there exists a subset $R \subseteq e_t$, $b(R) \geq 2^{r+1} \cdot c(e_t)$, such that $\text{eff}_{t+1}(v) \leq r$ for every $v \in R$. Since the effectiveness is non-decreasing, it follows that $\text{eff}_t(v) \leq r$ for every $v \in R$, hence the assumption that $b(R) \geq 2^{r+1} \cdot c(e_t)$ ensures that R is effective at time t. But by Obs. 2.7, the effectiveness $\text{eff}_{t+1}(v)$ should have been at least $r + 1$ for every $v \in R$, in contradiction to the choice of R. □

In accordance with the notation defined in Sec. 2.1.2, let $\text{eff}_\infty(v)$ denote the value of the variable $\text{eff}(v)$ upon termination of the input stream. Given some integer r, define

$$I(r) = \{v \in V \mid \text{eff}_\infty(v) = r\} \quad \text{and} \quad S(r) = \{e \in E \mid \exists v \in I(r) \text{ s.t. } \text{eid}(v) = \text{id}(e)\} .$$

We further extend these two definitions to intervals of integers in the natural way, and denote the intervals $(-\infty, r]$ and (r, ∞) in this context by $\leq r$ and $> r$, respectively. Thus, we use below the notations $I(\leq r)$ and $S(> r)$ where $I(\leq r) = \{v \in V \mid \text{eff}_\infty(v) \leq r\}$ and $S(> r) = \{e \in E \mid \exists v \in I(> r) \text{ s.t. } \text{eid}(v) = \text{id}(e)\}$.

LEMMA 2.9. *Procedure COVER guarantees that*

$$b(I(\leq r)) < 2^{r+1} \cdot c(\texttt{OPT}) .$$

PROOF. Since the effectiveness is non-decreasing, Lem. 2.8 ensures that for every edge $e \in E$, it holds that

$$b\left(\{v \in e \mid \text{eff}_\infty(v) \leq r\}\right) < 2^{r+1} \cdot c(e) .$$

The assertion is established by observing that

$$b(I(\leq r)) \leq \sum_{e \in \texttt{OPT}} b\left(\{v \in e \mid \text{eff}_\infty(v) \leq r\}\right) < \sum_{e \in \texttt{OPT}} 2^{r+1} \cdot c(e) = 2^{r+1} \cdot c(\texttt{OPT}) ,$$

where the first inequality is due to the fact that OPT is an edge cover of G. □

Lem. 2.9 will be used to bound from above the benefit of the vertices that are not covered by the edges returned by our algorithm. We now turn to bound from above the cost of these edges.

LEMMA 2.10. *Consider some integer r. The edge collection $S(r)$ satisfies*

$$c(S(r)) < b(V)/2^{r-1}\,.$$

PROOF. If $e_t \in S(r)$, then there exists some subset $R = R(e_t) \subseteq e_t$ with $\mathrm{lev}_t(R) = r$ such that for every vertex $v \in R$, we have (1) $\mathrm{eff}_t(v) < r$; and (2) $\mathrm{eff}_{t+1}(v) = r$. By definition, the fact that $\mathrm{lev}_t(R) = r$ implies that $c(e_t) < b(R)/2^{r-1}$. Since the variable $\mathrm{eid}(v)$ is updated only when $\mathrm{eff}(v)$ increases and since $\mathrm{eff}(v)$ is non-decreasing, it follows that if $e_t, e_{t'} \in S(r)$, $e_t \neq e_{t'}$, then the subsets $R(e_t)$ and $R(e_{t'})$ are disjoint. Therefore,

$$\sum_{e_t \in S(r)} c(e_t) < \frac{1}{2^{r-1}} \sum_{e_t \in S(r)} b(R(e_t)) \leq b(V)/2^{r-1}$$

which completes the proof. □

The following corollary is obtained by applying Lem. 2.10 to the integers $r+1, r+2, \ldots$

COROLLARY 2.11. *Consider some integer r. The edge collection $S(> r)$ satisfies*

$$c(S(> r)) < b(V)/2^{r-1}\,.$$

The following crucial lemma shows that we can extract from the variables returned by COVER an edge subset of low total cost which covers many of the items.

LEMMA 2.12. *Consider some $0 < \epsilon < 1$ and let r^* be the largest integer such that $b(I(\leq r^*)) \leq \epsilon \cdot b(V)$. The edge collection $S(> r^*)$ satisfies*

$$c(S(> r^*)) < 8 \cdot c(\mathtt{OPT})/\epsilon\,.$$

PROOF. Let r be an integer such that $2^{r+1} < \epsilon \cdot \frac{b(V)}{c(\mathtt{OPT})} \leq 2^{r+2}$. Lem. 2.9 guarantees that $b(I(\leq r)) < 2^{r+1} \cdot c(\mathtt{OPT}) < \epsilon \cdot b(V)$, hence $r \leq r^*$. It follows by Cor. 2.11 that $c(S(> r^*)) \leq c(S(> r)) < b(V)/2^{r-1} \leq 8 \cdot c(\mathtt{OPT})/\epsilon$. □

We are now ready to establish the approximation guarantees of algorithm SSSC. Theorem 2.1 (stated under the assumption that all vertex benefits and edge costs are encoded using $O(\log n)$ bits) follows immediately from Theorem 2.13.

THEOREM 2.13. *For any $0 \leq \epsilon < 1$, our algorithm outputs a $(1-\epsilon)$-cover certificate for G whose image has cost $O\left(\min\left\{\frac{1}{\epsilon}, \sqrt{n}\right\} \cdot c(\mathtt{OPT})\right)$.*

PROOF. If $\epsilon \geq 1/\sqrt{n}$, then the assertion follows immediately from Lem. 2.12, so it remains to consider the case of $\epsilon < 1/\sqrt{n}$. We show that χ'' is a 1-cover certificates for G such that $c(\mathrm{Im}(\chi'')) = O(\sqrt{n} \cdot c(\mathtt{OPT}))$. Observe first that since OPT covers all vertices in V, it is also an optimal edge 1-cover of G^1. Thus, Lem. 2.12 guarantees that $c(\mathrm{Im}(\chi')) < 8\sqrt{n} \cdot c(\mathtt{OPT})$. The vertices $v \in V - \mathrm{Dom}(\chi')$ are mapped under χ'' to $\mathrm{emin}(v)$. Since $|V - \mathrm{Dom}(\chi')| \leq \sqrt{n}$ and since $c(\mathrm{emin}(v)) \leq c(\mathtt{OPT})$ for every $v \in V$, it follows that

$$c(\mathrm{Im}(\chi'')) < 8\sqrt{n} \cdot c(\mathtt{OPT}) + |V - \mathrm{Dom}(\chi')| \cdot c(\mathtt{OPT}) \leq 9\sqrt{n} \cdot c(\mathtt{OPT})\,.$$

The assertion follows. □

2.3. Lifting the assumption on the numerical values

We now turn to lift the assumption that all numerical values are encoded using $O(\log n)$ bits and establish Thm. 2.2 and Thm. 2.3, starting with the former. To that end, consider the hypergraph $\widetilde{G} = (V, E, \widetilde{b}, \widetilde{c})$ defined by setting $\widetilde{b}(v) = 2^{\lfloor \log b(v) \rfloor}$ for every vertex $v \in V$ and $\widetilde{c}(e) = 2^{\lfloor \log c(e) \rfloor}$ for every edge $e \in E$. Since $\widetilde{b}(U)$ and $\widetilde{c}(F)$ are 2-approximations of $b(U)$ and $c(F)$, respectively, for every $U \subseteq V$ and $F \subseteq E$, it follows that a $(1-O(\epsilon))$-cover certificate for G with image of cost $O\left(\min\left\{\frac{1}{\epsilon}, \sqrt{n}\right\} \cdot c(\mathtt{OPT})\right)$ can be obtained by running SSSC on $\widetilde{G}$.

So, in what follows, we assume that $b(v)$ and $c(e)$ are (not necessarily positive) integral powers of 2 for every vertex $v \in V$ and edge $e \in E$. This implies that every benefit $b(v)$ (resp., cost $c(e)$) in G can be encoded using $O(\log b^{\log})$ (resp., $O(\log c^{\log})$) bits simply by taking the standard binary representation of $\log b(v)$ (resp., $\log c(e)$) (see the beginning of Sec. 2 for the definition of $b^{\log}$ and $c^{\log}$). Therefore, procedures P3 and P4 can be implemented using $O\left(\log\left(n + m + b^{\log} + c^{\log}\right)\right)$ bits per vertex, as desired. Procedure COVER can also be implemented with that many bits per vertex since the level at time t of each subset $T \subseteq e_t$ is an integer whose absolute value satisfies $|\mathrm{lev}_t(T)| = O(b^{\log} + c^{\log} + \log n)$, thus establishing Thm. 2.2 due to Obs. 2.6 and Thm. 2.13.

For Thm. 2.3, we need two additional features. First, we scale in an online fashion all vertex benefits and edge costs so that $\min_{v \in V} b(v)$ and $\min_{e \in E} c(e)$ are always 1. We do the same thing with the effectiveness variables $\mathrm{eff}(v)$, only that, this time, we ignore those variables with $\mathrm{eff}(v) = -\infty$. This is carried out by maintaining the true values of $\min_{v \in V} b(v)$, $\min_{e \in E} c(e)$, and $\min_{v \in V : \mathrm{eff}(v) > -\infty} \mathrm{eff}(v)$ — denote them by $b_{\min}$, $c_{\min}$, and $\mathrm{eff}_{\min}$, respectively — and scaling all values of $b(v)$, $c(e)$, and $\mathrm{eff}(v)$, stored in the data structures maintained by the procedures of our algorithm, by $b_{\min}$, $c_{\min}$, and $\mathrm{eff}_{\min}$, respectively. Notice that this online scaling requires updating the existing values stored in the data structures whenever $b_{\min}$, $c_{\min}$, or $\mathrm{eff}_{\min}$ are updated, thus resulting in the less favorable run-time promised in Thm. 2.3.

This online scaling feature ensures that the space needed for the variables of each vertex v is now

$$O\left(\log\left(n + m + b^{\Delta} + c^{\Delta}\right)\right) , \tag{1}$$

where $b^{\Delta} = \log\left\lceil \frac{\max_{v \in V} b(v)}{\min_{v \in V} b(v)} \right\rceil$ is the number of bits required to encode the vertex benefits aspect ratio. We also need additional $O(\log(b^{\log} + c^{\log}))$ bits to store the variables $b_{\min}$, $c_{\min}$, and $\mathrm{eff}_{\min}$.

In order to get rid of the dependency on $\log b^{\Delta}$ in (1) and obtain the space bound promised in Thm. 2.3, we use the following feature: Let $\sigma = \sum_{v \in V'} b(v)$, where V' is the set of vertices $v \in V$ encountered by the algorithm so far. Whenever it becomes clear that the contribution of some vertex $v \in V$ to $b(V)$ is at most $\epsilon \cdot b(V)/n$, which is indicated by $b(v) \leq \epsilon\sigma/n$, the algorithm marks vertex v as *insignificant*. Insignificant vertices are treated as if they are not part of the input hypergraph G; in particular, upon marking vertex v as insignificant, the algorithm erases any variable associated with v and updates $b_{\min}$ so that it does not take $b(v)$ into account.

Notice that the total contribution of all insignificant vertices to $b(V)$ is bounded from above by $\epsilon \cdot b(V)$. Therefore, ignoring insignificant vertices cannot decrease our guaranteed coverage by more than an additive term of $\epsilon \cdot b(V)$. The key observation now is that by ignoring insignificant vertices, we keep the parameter b^{Δ} bounded by $b^{\Delta} = O(\log(n/\epsilon))$ as the benefit of any vertex encountered by the algorithm so far is clearly at most σ. Recalling that ϵ is always at least $1/\sqrt{n}$, we conclude that the dependency on $\log b^{\Delta}$ in (1) is replaced by a dependency on $\log\log n$. Thm. 2.3 now follows by Thm. 2.13.

3. LOWER BOUNDS

We start with a number of definitions that simplify the statements of our lower bounds. A *randomized* semi-streaming algorithm ALG for the edge cover problem in hypergraphs is said to be an (n, s, ϵ, ρ)-*algorithm* (resp., an *uncertified* (n, s, ϵ, ρ)-*algorithm*) if given any n-vertex unweighted hypergraph G, ALG is guaranteed to maintain a memory of size at most s bits and to output a $(1-\epsilon)$-cover certificate for G with image of expected cardinality at most $\rho \cdot |\texttt{OPT}|$ (resp., to output the identifiers of an edge $(1-\epsilon)$-cover of G whose expected size is at most $\rho \cdot |\texttt{OPT}|$), where $\texttt{OPT}$ is an optimal edge cover of G. Our goal in this section is to establish Thm. 3.1 and Thm. 3.2, proved in Sec. 3.1 and Sec. 3.2, respectively. Observe that the constructions that lie at the heart of Theorems 3.1 and 3.2 are based on hypergraphs whose number of vertices and number of edges are polynomially related, that is, $m = n^{\Theta(1)}$.

THEOREM 3.1. *For every integer n_0, there exists an integer $n \geq n_0$ such that for every $\epsilon = \Omega(1/\sqrt{n})$, the existence of an $(n, o(n^{3/2}), \epsilon, \rho)$-algorithm implies that $\rho = \Omega(1/\epsilon)$.*

THEOREM 3.2. *Fix some constant real $\alpha > 0$. For every integer n_0, there exists an integer $n \geq n_0$ such that for every $\epsilon \geq n^{-1/2+\alpha}$, the existence of an uncertified $(n, o(n^{1+\alpha}), \epsilon, \rho)$-algorithm implies that $\rho = \Omega\left(\frac{\log\log n}{\log n}\frac{1}{\epsilon}\right)$.*

3.1. The certified case

We shall establish Thm. 3.1 by introducing a probability distribution $\mathcal{G}$ over n-vertex hypergraphs that satisfy the following two properties: (1) Every hypergraph in the support of $\mathcal{G}$ admits an edge cover of cardinality $O(\epsilon\sqrt{n})$. (2) For every *deterministic* semi-streaming algorithm $\texttt{ALG}$ that, given an n-vertex hypergraph G, maintains a memory of size $o(n^{3/2})$ and outputs a $(1-\epsilon)$-cover certificate χ for G, when $\texttt{ALG}$ is invoked on a hypergraph chosen according to $\mathcal{G}$, the expected cardinality of $\mathrm{Im}(\chi)$ is $\Omega(\sqrt{n})$. The theorem then follows by Yao's principle.

3.1.1. The construction of $\mathcal{G}$. Let q be a large prime power. Our construction relies on the *affine plane* $\mathcal{A} = (P, L)$, where P is a set of q^2 *points* and $L \subseteq 2^P$ is a set of $q(q+1)$ *lines* satisfying the following properties:
(1) every line contains q points;
(2) every point is contained in $q+1$ lines;
(3) for every two distinct points, there is exactly one line that contains both of them; and
(4) every two lines intersect in at most one point.
Two lines with an empty intersection are called *parallel*. The line set L can be partitioned into $q+1$ clusters $A_1, \dots, A_{q+1}$, referred to as *angles*, where $A_i = \{\ell_i^1, \dots, \ell_i^q\}$ for $i = 1, \dots, q+1$, such that two distinct lines are parallel if and only if they belong to the same angel. For example, Refer to [Lindner and Rodger 2011] for an explicit construction of such a combinatorial structure.

Consider some $\frac{1}{3q} \leq \epsilon \leq \frac{1}{66} - \frac{1}{3q}$ and let $r = \lceil 3\epsilon q \rceil$. We construct a random hypergraph $G = (V, E)$ based on the affine plane $\mathcal{A} = (P, L)$ as follows (refer to Figure 1 for an illustration). Fix $V = P$. Randomly partition each line $\ell \in L$ into 2 edges $e_1(\ell) \cup e_2(\ell) = \ell$ by assigning each point in L to one of the 2 edges u.a.r. (and independently of all other random choices).[5] It will be convenient to denote the set of edges corresponding to the lines in angle A_i by $E_i = \{e_1(\ell), e_2(\ell) \mid \ell \in A_i\}$. Let

$$e^* = P - \bigcup_{t=1}^{r} \ell_i^{j(t)},$$

where i is an index chosen u.a.r. (and independently of all other random choices) from $[q+1]$, and $1 \leq j(1) < \cdots < j(r) \leq q$ are r distinct indices chosen u.a.r. (and independently of all other random choices) from $[q]$. In other words, e^* is constructed by ran-

[5]Throughout, we use u.a.r. to abbreviate "uniformly at random".

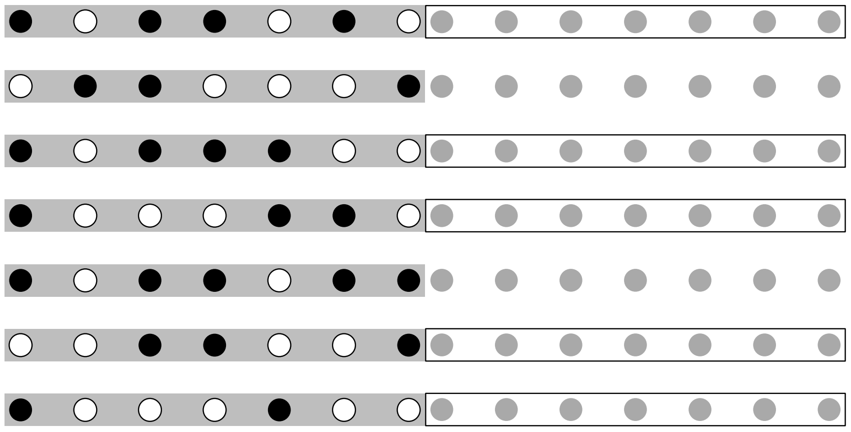

Fig. 1: The hypergraph G for $q = 7$. (The requirements on ϵ actually imply that q must be larger, but we set $q = 7$ for the sake of a clearer illustration.) The gray rectangles in (a) depict the 7 parallel lines in angle A_i for some $i \in [q + 1]$, whereas the black/white circles in each line ℓ_i^j depict the points in $e_1(\ell_i^j)$/$e_2(\ell_i^j)$. Edge e^*, depicted by the white rectangles in (b), consists of all points except those in $r = 2$ lines of angle A_i.

domly choosing an angle A_i and then randomly choosing r distinct lines $\ell_i^{j(1)}, \dots, \ell_i^{j(r)}$ from A_i; the edge e^* consists of all points except those contained in these r lines.

Fix

$$E = E_1 \cup \cdots \cup E_{q+1} \cup \{e^*\} .$$

Observe that $n = |P| = q^2$ and $m = 1+2\cdot|L| = 1+2\cdot q(q+1)$. The input stream is divided into two stages, where in the first stage the edges in $E_1 \cup \cdots \cup E_{q+1}$ are presented in an arbitrary order, and in the second stage, edge e^* is presented.

3.1.2. Analysis. We start the analysis by observing that G can be covered by the edge e^* and the edges in $\{e_1(\ell_i^{j(t)}), e_2(\ell_i^{j(t)}) \mid 1 \leq t \leq r\}$. Therefore,

$$|\mathtt{OPT}| \leq 2r + 1 = O(\epsilon q) , \tag{2}$$

where the equation follows from the definition of $r = \lceil 3\epsilon q \rceil$ and the requirement that $\epsilon \geq \frac{1}{3q}$.

Let s be the size of the space used by the deterministic semi-streaming algorithm ALG. Thm. 3.1 is established by combining (2) with the following lemma that ensures an $\Omega(q)$ expected image cardinality whenever $s = o(n^{3/2})$.

LEMMA 3.3. *If $s \leq q^2(q+1)/48$, then w.p. $\geq 1/8$, the $(1-\epsilon)$-cover certificate returned by* ALG *has image of cardinality at least $q/3$.*[6]

The remainder of this section is dedicated to proving Lem. 3.3 based on the following information theoretic arguments.

[6]Throughout, we use w.p. and w.h.p. to abbreviate "with probability" and "with high probability", respectively.

Bounding from below the expected entropy. Let X_i^j be a random variable that depicts the partition $(e_1(\ell_i^j), e_2(\ell_i^j))$ of line $\ell_i^j = e_1(\ell_i^j) \cup e_2(\ell_i^j)$ for every $i \in [q+1]$ and $j \in [q]$. Let $X_i = (X_i^1, \ldots, X_i^q)$ and $X = (X_1, \ldots, X_{q+1})$. The independent random choices in the construction of the hypergraph G guarantee that $H(X_i^j) = q$, $H(X_i) = q^2$, and $H(X) = q^2(q+1)$, where $H(\cdot)$ denotes the entropy function.[7] Before we can proceed with our proof, we have to establish the following lemma whose restriction to the case $k = 1$ is a basic fact in information theory. It will not strike us as a surprise if this lemma was already proved beforehand although we are unaware of any such specific proof; for the sake of completeness, we provide in Appendix A a full proof of this lemma based on Baranyai's Theorem.

LEMMA 3.4. *Let $X_1, \ldots, X_n, Y$ be $n+1$ arbitrary random variables and let $1 \leq j(1) < \cdots < j(k) \leq n$ be $1 \leq k \leq n$ distinct indices chosen u.a.r. from $[n]$. Then,*

$$\left\lceil \frac{n}{k} \right\rceil \mathbb{E}_{j(1),\ldots,j(k)} \left[H\left(X_{j(1)}, \ldots, X_{j(k)} \mid Y \right) \right] \;\geq\; H\left(X_1, \ldots, X_n \mid Y \right) .$$

Let M be a random variable that depicts the memory image of ALG upon completion of the first stage of the input stream. Since M is fully determined by X, it follows that $H(X, M) = H(X)$, hence $H(X \mid M) = H(X) - H(M)$. Recalling that M is described by s bits, we conclude that $H(M) \leq s \leq q^2(q+1)/48$, thus

$$H(X \mid M) \geq \frac{47}{48} \cdot q^2(q+1) = \frac{47}{48} \cdot H(X) . \tag{3}$$

We are now ready to establish the following lemma.

LEMMA 3.5. *Our construction guarantees that*

$$\mathbb{P}_{i,j(1),\ldots,j(r)} \left(H\left(X_i^{j(1)}, \ldots, X_i^{j(r)} \mid M \right) \geq \frac{5}{6} \cdot rq \right) \geq 1/4 ,$$

where $i \in [q+1]$ and $1 \leq j(1) < \cdots < j(r) \leq q$ are the random indices chosen during the construction of edge e^*.

PROOF. By combining (3) with an application of Lem. 3.4 to the random choice of index $i \in [q+1]$, we derive the inequality

$$\mathbb{E}_i \left[H\left(X_i \mid M \right) \right] \geq \frac{47}{48} \cdot q^2 .$$

Since $H(X_i \mid M) \leq q^2$, we can apply Markov's inequality to conclude that

$$H(X_i \mid M) \geq \frac{23}{24} \cdot q^2 \tag{4}$$

w.p. $\geq 1/2$.

Conditioned on the event that (4) holds, we can apply Lem. 3.4 to the random choice of indices $1 \leq j(1) < \cdots < j(r) \leq q$, deriving the inequality

$$\left\lceil \frac{q}{r} \right\rceil \mathbb{E}_{j(1),\ldots,j(r)} \left[H\left(X_i^{j(1)}, \ldots, X_i^{j(r)} \mid M \right) \right] \geq \frac{23}{24} \cdot q^2 ,$$

which means that

$$\mathbb{E}_{j(1),\ldots,j(r)} \left[H\left(X_i^{j(1)}, \ldots, X_i^{j(r)} \mid M \right) \right] \geq \frac{23}{24} \frac{rq^2}{q+r} .$$

[7]The entropy of a discrete random variable Z is defined to be $H(Z) = -\sum_z \mathbb{P}(Z = z) \cdot \log \mathbb{P}(Z = z)$, where the summation runs over all values z in the support of Z.

Since $\epsilon \leq \frac{1}{66} - \frac{1}{3q}$, it follows that $r = \lceil 3\epsilon q \rceil \leq 3\epsilon q + 1 \leq q/22$. This, in turn, implies that $\frac{23}{24} \frac{rq^2}{q+r} \geq \frac{11}{12} rq$, hence

$$\mathbb{E}_{j(1),\dots,j(r)} \left[H\left(X_i^{j(1)}, \dots, X_i^{j(r)} \mid M \right) \right] \geq \frac{11}{12} \cdot rq \, .$$

Since $H(X_i^{j(1)}, \dots, X_i^{j(r)} \mid M) \leq rq$, we can apply Markov's inequality to conclude that

$$H\left(X_i^{j(1)}, \dots, X_i^{j(r)} \mid M \right) \geq \frac{5}{6} \cdot rq$$

w.p. $\geq 1/2$. The assertion follows as (4) holds w.p. $\geq 1/2$. □

Introducing the random variable Z. Let $i \in [q+1]$ and $1 \leq j(1) < \cdots < j(r) \leq q$ be the random indices chosen during the construction of edge e^*. Let μ be the actual memory image of ALG upon completion of the first stage of the input stream, and recall that μ is some instantiation of the random variable M. Let Z be a real valued random variable that maps the event $M = \mu$ to the entropy in the joint random variable $X_i^{j(1)}, \dots, X_i^{j(r)}$, given $M = \mu$; in other words, $Z(\mu) = H(X_i^{j(1)}, \dots, X_i^{j(r)} \mid M = \mu)$.[8] Observe that by the definition of conditional entropy, we have $\mathbb{E}[Z] = H(X_i^{j(1)}, \dots, X_i^{j(r)} \mid M)$. If the event described in the statement of Lem. 3.5 occurs, then $\mathbb{E}[Z] \geq \frac{5}{6} \cdot rq$ and since Z is never larger than rq, we can apply Markov's inequality to conclude that

$$\mathbb{P}\left(Z \geq \frac{2}{3} \cdot rq \right) \geq 1/2 \, .$$

The following corollary is established since the event described in Lem. 3.5 holds w.p. $\geq 1/4$.

COROLLARY 3.6. *W.p. $\geq 1/8$, the entropy that remains in $X_i^{j(1)}, \dots, X_i^{j(r)}$ after e^* is exposed to ALG, given that $M = \mu$, is at least $\frac{2}{3} \cdot rq$ bits.*

High entropy implies a large expected edge cover. Condition hereafter on the event described in the statement of Cor. 3.6. Consider the $(1-\epsilon)$-cover certificate χ returned by ALG and let $P' = \bigcup_{t=1}^{r} \ell_i^{j(t)} = P - e^*$ be the set of points not covered by e^*. Let

$$R = \{ p \in P' \mid p \in \mathrm{Dom}(\chi) \wedge \chi(p) \in E_i \}$$

be the set of points, not covered by e^*, that are mapped under χ to some edge in E_i (recall that E_i is the set of edges corresponding to the lines in angle A_i, i.e., the angle chosen in the random construction of e^*). We can now establish the following lemma.

LEMMA 3.7. *Our construction guarantees that $|R| \leq rq/3$.*

PROOF. The joint random variable $X_i^{j(1)}, \dots, X_i^{j(r)}$ conditioned on $M = \mu$ can be viewed as a probability distribution π over the matrices $T \in \{1,2\}^{r \times q}$, where $T(t,k) \in \{1,2\}$ indicates whether the kth point in line $\ell_i^{j(t)}$ belongs to edge $e_1(\ell_i^{j(t)})$ or to edge $e_2(\ell_i^{j(t)})$ for every $k \in [q]$ and $1 \leq t \leq r$.

Consider some point $p \in P'$ and suppose that this is the kth point in line $\ell_i^{j(t)}$. The key observation is that if $p \in R$, then all matrices T in the support of π must agree on

[8]Note that the notation $H(X_i^{j(1)}, \dots, X_i^{j(r)} \mid \mu)$ is sometimes used in the literature for the same quantity.

$T(t,k)$.[9] Therefore, the entropy that remains in $X_i^{j(1)}, \ldots, X_i^{j(r)}$ after e^* is exposed to ALG can only arrive from points in $P' - R$. The assertion follows by Cor. 3.6 since each such point contributes at most 1 bit of entropy. $\square$

The cardinality of $\mathrm{Dom}(\chi)$ is at least $|\mathrm{Dom}(\chi)| \geq (1-\epsilon)q^2$. The choice of $r = \lceil 3\epsilon q \rceil$ ensures that $\epsilon q^2 \leq rq/3$, thus $|\mathrm{Dom}(\chi)| \geq q^2 - rq/3$. The key observation now is that even if all these $rq/3$ missing points from $\mathrm{Dom}(\chi)$ are in P', it still leaves us with $|\mathrm{Dom}(\chi) \cap (P' - R)| \geq rq/3$ by Lem. 3.7.

Every point in $\mathrm{Dom}(\chi) \cap (P' - R)$ is covered by some edge $e \in E_j$, $j \neq i$. The properties of the affine plane guarantee that each such edge e covers at most one point in line $\ell_i^{j(t)}$, which sums up to at most r points in P'. Thus, the image of χ must contain (the identifiers of) at least $q/3$ different edges. This concludes the proof of Lem. 3.3. Thm. 3.1 then follows by combining (2) and Lem. 3.3.

3.2. The uncertified case

Similarly to the proof of Thm. 3.1, we shall establish Thm. 3.2 by introducing a probability distribution $\mathcal{G}'$ over n-vertex hypergraphs that, this time, satisfies the following two properties: (1) every hypergraph in the support of $\mathcal{G}'$ admits an edge cover of cardinality $O(\epsilon^2 n)$; and (2) for every *deterministic* semi-streaming algorithm ALG that, given an n-vertex hypergraph $G = (V, E)$, maintains a memory of size $o(n^{1+\alpha})$ and outputs the identifiers of an edge $(1-\epsilon)$-cover $F \subseteq E$ of G, when ALG is invoked on a hypergraph chosen according to $\mathcal{G}'$, the expected cardinality of F is $\Omega\left(\epsilon n \frac{\log \log n}{\log n}\right)$. The theorem then follows by Yao's principle.

3.2.1. The construction of $\mathcal{G}'$. We construct a random hypergraph $\hat{G} = (\hat{V}, \hat{E})$ as follows. Let q be a large power of 2 and fix some real constant $\alpha > 0$. Consider some $q^{-(1-\alpha)} \leq \epsilon \leq \frac{1}{66} - \frac{1}{3q}$ and let $r = \lceil 3\epsilon q \rceil$. The main building block of $\hat{G}$ is very similar to the random hypergraph $G = (V, E)$ constructed in Sec. 3.1.1 based on the affine plane $\mathcal{A} = (P, L)$. Specifically, fix $\hat{V} = P$ and let E' be a random edge set constructed just like the construction of the random edge set E presented in Sec. 3.1.1 with the following exception: Instead of randomly partitioning each line $\ell \in L$ into 2 edges $e_1(\ell) \cup e_2(\ell) = \ell$ by assigning each point in ℓ to one of the 2 edges u.a.r. (and independently), we randomly partition each line $\ell \in L$ into r edges $e_1(\ell) \cup \cdots \cup e_r(\ell) = \ell$ by assigning each point in ℓ to one of the r edges u.a.r. (and independently).

The edge e^* is constructed in the same manner as in Sec. 3.1.1, i.e., we choose an angle A_i u.a.r. and then choose r distinct lines $\ell_i^{j(1)}, \ldots, \ell_i^{j(r)}$ u.a.r. from A_i; the edge e^* consists of all points except those contained in these r lines. Notice that the parameter r is now used for both the partition of each line into r edges and the construction of edge e^*. For every $i \in [q+1]$, denote the set of edges corresponding to the lines in angle A_i by $E'_i = \{e_1(\ell), \ldots, e_r(\ell) \mid \ell \in A_i\}$ and fix $E' = E'_1 \cup \cdots \cup E'_{q+1} \cup \{e^*\}$.

The edge multi-set $\hat{E}$ is obtained from E' by augmenting it with *dummy* edges: fix $\hat{E} = E' \cup E_{\mathrm{d}}$, where the dummy edges, i.e., $e \in E_{\mathrm{d}}$, are all empty $e = \emptyset$. (Concerns regarding the usage of empty edges can be lifted by augmenting $\hat{V}$ with a dummy vertex v_{d} and taking all dummy edges $e \in E_{\mathrm{d}}$ to be singletons $e = \{v_{\mathrm{d}}\}$.)

[9]In fact, even if we relax the requirement from ALG so that χ is allowed to err on some vertices in its domain but the coverage is measured only with respect to the vertices for which χ is correct, we can still achieve the desired (asymptotic) bound by using a line of arguments similar to that used in the proof of Lemma 6.2 in [Alon et al. 2013].

Identifier assignment. Recall that the identifiers of the edges are determined by their arrival order so that the edge e_t arriving at time t is assigned with identifier $\mathrm{id}(e_t) = t$. In contrast to the construction presented in Sec. 3.1.1, where the identifier assignment is arbitrary (with the exception that $\mathrm{id}(e^*)$ should be the largest identifier), the assignment of identifiers to the edges in $\hat{E}$ plays a key role in the current construction. Specifically, for every $i \in [q+1]$, $j \in [q]$, and $k \in [r]$, the identifier assigned to edge $e_k(\ell_i^j)$ is

$$\mathrm{id}(e_k(\ell_i^j)) = 0 \circ i \circ j \circ k \circ X_i^{j,k} ,$$

where i, j, and k are encoded as bitstrings of lengths $\lceil \log(q+1) \rceil$, $\log q$ (recall that q is a power of 2), and $\lceil \log r \rceil$, respectively, $\circ$ denotes the string concatenation operator, and $X_i^{j,k}$ is a bitstring of length $3 \log q$ chosen u.a.r. (and independently). Notice that each identifier contains $\iota = 1 + \lceil \log(q+1) \rceil + \log q + \lceil \log r \rceil + 3 \log q$ bits encoding some integer (with the most significant bit on the left) in $[0, 2^{\iota-1} - 1]$ and, by design, each edge in $E'_1 \cup \cdots \cup E'_{q+1}$ is assigned with a unique identifier.

The identifier assigned to edge e^* is $\mathrm{id}(e^*) = 1 \circ 0^{\iota-1}$, which encodes the integer $2^{\iota-1}$. The dummy edges are used for filling up the gaps between the identifiers assigned to the edges in E' so that $\mathrm{id}(\cdot)$ is a bijection from $\hat{E} = E' \cup E_{\mathrm{d}}$ to $\left[0, 2^{\iota-1}\right]$. As e^* is assigned with the highest identifier, this is the last edge to arrive. Observe that $n = q^2$ and $m = 2^{\iota-1} + 1 = O(q^6)$.

3.2.2. Analysis. We start the analysis by observing that $\hat{G}$ can be covered by edge e^* and the edges in $\{e_1(\ell_i^{j(t)}), \ldots, e_r(\ell_i^{j(t)}) \mid 1 \leq t \leq r\}$. Therefore,

$$|\mathtt{OPT}| \leq r^2 + 1 = O(\epsilon^2 q^2) , \tag{5}$$

where the equation follows from the definition of $r = \lceil 3\epsilon q \rceil$ and the requirement that $\epsilon = \omega(q^{-1})$.

Let s be the size of the space used by the deterministic semi-streaming algorithm $\mathtt{ALG}$. Thm. 3.2 is established by combining (5) with the following lemma that ensures an $\tilde{\Omega}(\epsilon q^2)$ expected set cover cardinality whenever $s = o(\epsilon n^{3/2})$.

LEMMA 3.8. *If $s \leq rq(q+1)/16$, then w.p. $\geq 1/9$, the edge $(1-\epsilon)$-cover returned by* $\mathtt{ALG}$ *has cardinality* $\Omega\left(\epsilon q^2 \frac{\log \log q}{\log q}\right)$.

The remainder of this section is dedicated to proving Lem. 3.8 based on the following information theoretic arguments. Recall that $X_i^{j,k}$ is a random bitstring of length $3 \log q$ used in the construction of $\mathrm{id}(e_k(\ell_i^j))$ for every $i \in [q+1]$, $j \in [q]$, and $k \in [r]$. Let $X_i^j = (X_i^{j,1}, \ldots, X_i^{j,r})$, $X_i = (X_i^1, \ldots, X_i^q)$, and $X = (X_1, \ldots, X_{q+1})$. The independent random choices in the construction of the identifiers of $\hat{E}$ guarantee that $H(X_i^{j,k}) = 3 \log q$, $H(X_i^j) = 3r \log q$, $H(X_i) = 3rq \log q$, and $H(X) = 3rq(q+1) \log q$.

As in the analysis in Sec. 3.1.2, let $i \in [q+1]$ and $1 \leq j(1) < \cdots < j(r) \leq q$ be the random indices chosen in the construction of edge e^*. Let M be a random variable that depicts the memory image of $\mathtt{ALG}$ before the last edge e^* arrives and let μ be its actual instantiation. Observing that $H(X \mid M) \geq \frac{47}{48} \cdot H(X)$ (cf. inequality (3)), we can repeat the line of arguments used in Sec. 3.1.2 to derive the following corollary (analogous to Cor. 3.6).

COROLLARY 3.9. *W.p. $\geq 1/8$, the entropy that remains in $X_i^{j(1)}, \ldots, X_i^{j(r)}$ after e^* is exposed to* $\mathtt{ALG}$*, given that $M = \mu$, is at least $2r^2 \log q$ bits.*

Notice that the requirement $\epsilon \geq q^{-(1-\alpha)}$ ensures that $r = \lceil 3\epsilon q \rceil$ and q are polynomially related and so are r and $n = q^2 + 1$. Therefore, an event that holds w.h.p. with respect to the parameter r also holds w.h.p. with respect to the parameters q and n; in what follows, whenever we use the term w.h.p., we refer to w.h.p. with respect to these three parameters.

LEMMA 3.10. *W.h.p., all edges $e_k(\ell_i^{j(t)})$, $t \in [r], k \in [r]$, satisfy $(5/6)q/r \leq |e_k(\ell_i^{j(t)})| \leq 2q/r$.*

PROOF. Fix some $t \in [r]$ and $k \in [r]$. The random partition of line $\ell_i^{j(t)}$ into the r edges $e_1(\ell_i^{j(t)}) \cup \cdots \cup e_r(\ell_i^{j(t)}) = \ell_i^{j(t)}$ implies that $\mathbb{E}[|e_k(\ell_i^{j(t)})|] = q/r$. By Chernoff's bound, we have $(5/6)q/r \leq |e_k(\ell_i^{j(t)})| \leq 2q/r$ w.h.p. The assertion follows by the union bound. $\square$

Identifiers with large entropy. Condition hereafter on the events described in the statements of Cor. 3.9 and of Lem. 3.10. Since Cor. 3.9 ensures that

$$\sum_{t=1}^{r}\sum_{k=1}^{r} H(X_i^{j(t),k} \mid M = \mu) \geq H(X_i^{j(1)}, \ldots, X_i^{j(r)} \mid M = \mu) \geq 2r^2 \log q \ ,$$

and since $H(X_i^{j(t),k} \mid M = \mu) \leq 3 \log q$ for every $(t,k) \in [r] \times [r]$, it follows that there exists a subset $\Psi \subseteq [r] \times [r]$ such that (1) $|\Psi| \geq r^2/2$; and (2) $H(X_i^{j(t),k} \mid M = \mu) \geq \log q$ for every $(t,k) \in \Psi$.

Consider some pair $(t,k) \in \Psi$. The definition of Ψ guarantees that at least $\log q$ bits of entropy remain in the identifier $\mathrm{id}(e_k(\ell_i^{j(t)}))$ of edge $e_k(\ell_i^{j(t)})$ after e^* is exposed to `ALG` given that $M = \mu$. Thus, `ALG` must have at least q different candidates for $\mathrm{id}(e_k(\ell_i^{j(t)}))$. The design of the identifier assignment function $\mathrm{id}(\cdot)$ guarantees that all but one of these candidate identifiers are assigned to dummy edges and that the candidate identifiers of edge $e_k(\ell_i^{j(t)})$ and the candidate identifiers of edge $e_{k'}(\ell_i^{j(t')})$ are disjoint for every $(t,k), (t',k') \in \Psi$, $(t,k) \neq (t',k')$. Therefore, every edge $e_k(\ell_i^{j(t)})$ with $(t,k) \in \Psi$ that is guaranteed to belong to the edge $(1-\epsilon)$-cover F output by `ALG` contributes at least q distinct identifiers to the output of `ALG`.

On the other hand, Lem. 3.10 ensures (w.h.p.) that all points in each line $e_k(\ell_i^{j(t)})$ can be covered by at most $2q/r \ll q$ edges belonging to $E'_{-i} = E'_1 \cup \cdots \cup E'_{i-1} \cup E'_{i+1} \cup \cdots \cup E'_{q+1}$, that is, edges corresponding to lines of angles other than A_i. Hence, for the sake of proving the lower bound on the number of edges used, we may assume hereafter that, for every $(t,k) \in \Psi$, `ALG` covers the points in $e_k(\ell_i^{j(t)})$ by edges belonging to E'_{-i}.

Coverage from another angle. Let $N = \bigcup_{(t,k)\in\Psi} e_k(\ell_i^{j(t)})$ be the set of points contained in the edges corresponding to the index pairs in Ψ. Since $|\Psi| \geq r^2/2$ and since Lem. 3.10 guarantees that $|e_k(\ell_i^{j(t)})| \geq (5/6)q/r$ for every $(t,k) \in \Psi$, it follows that $|N| \geq 5qr/12$.

Recall that the edge $(1-\epsilon)$-cover F may leave at most ϵq^2 uncovered points. The choice of $r = \lceil 3\epsilon q \rceil$ ensures that $\epsilon q^2 \leq qr/3$, thus at most $qr/3$ points are not covered by F. The key observation now is that even if all these uncovered points belong to N, then F should still cover at least $5qr/12 - qr/3 = qr/12$ points in N; let $N' \subseteq N$ be the subset consisting of these (at least) $qr/12$ covered points.

We argue that in order to cover the points in N' with edges belonging to E_{-i}, one needs $\Omega\left(\epsilon q^2 \frac{\log\log q}{\log q}\right) = \Omega\left(qr \frac{\log\log q}{\log q}\right)$ distinct edges w.h.p. The proof of Lem. 3.8 is completed by the union bound since the events described in the statements of Cor. 3.9 and of Lem. 3.10 (i.e., the events on which our analysis is conditioned) hold w.p. $\geq 1/8$ and w.h.p., respectively. To that end, consider some line $\ell \in L - A_i$, namely, a line from an angle other than A_i. The properties of the affine plane $\mathcal{A}$ ensure that the intersection $I(\ell) = \ell \cap (\ell_i^{j(1)} \cup \cdots \cup \ell_i^{j(r)})$ contains exactly $|I(\ell)| = r$ points. The assignment of these r points to the edges $e_1(\ell), \ldots, e_r(\ell)$ is determined by the random partition of ℓ into $e_1(\ell) \cup \cdots \cup e_r(\ell) = \ell$ and it can be viewed as a balls-into-bins process with r balls and r bins. By a known result on balls-into-bins processes (see, e.g., [Mitzenmacher and Upfal 2005]), we conclude that w.h.p., $\max_{k \in [r]} |e_k(\ell) \cap I(\ell)| = O\left(\frac{\log r}{\log \log r}\right)$ and by the union bound, this holds for all lines $\ell \in L - A_i$ w.h.p.; in particular, every edge in E'_{-i} covers $O\left(\frac{\log r}{\log \log r}\right)$ points in N' . The argument follows since $|N'| = \Omega(qr)$.

This concludes the proof of Lem. 3.8. Thm. 3.2 then follows by combining (5) and Lem. 3.8.

4. CONCLUSIONS

We provide a deterministic $O(\min\{1/\epsilon, \sqrt{n}\})$-approximation for the edge $(1-\epsilon)$-cover problem in hypergraphs under the (single pass) semi-streaming model of computation. The algorithm maintains a data structure of size $O(n \log |G|)$ (where $|G|$ denotes the number of bits in the standard binary encoding of the input hypergraph G) from which the desired $(1-\epsilon)$-cover certificate can be extracted for any $0 \leq \epsilon < 1$. We prove that the tradeoff between the coverage parameter ϵ and the approximation ratio is asymptotically tight: single-pass algorithms with better approximation ratio must use $\Omega(n^{3/2})$ space. The algorithm is simple to implement and the hidden constants are relatively small, so it may be useful in practice.

The benchmark used in the present paper for the approximation algorithm is the optimal edge 1-cover even when $\epsilon > 0$. This leaves the following interesting open question: How well can one approximate the edge $(1-\epsilon)$-cover problem under the semi-streaming model when the benchmark is the optimal edge $(1-\epsilon)$-cover (which, in general, can be significantly smaller than the optimal edge 1-cover)?

APPENDIX

A. PROOF OF LEMMA 3.4

Assume first that $n/k = d$ for some integer $d \geq 1$. Let $\mathcal{S}(n, k)$ be the collection of all $\binom{n}{k}$ subsets $S \subseteq [n]$ of cardinality $|S| = k$. By Baranyai's Theorem (see, e.g., [van Lint and Wilson 2001]), there exists a partition $\mathcal{P}$ of $\mathcal{S}(n, k)$ into $\binom{n}{k}/d$ pairwise disjoint clusters such that every cluster C of $\mathcal{P}$ consists of d subsets $S \in \mathcal{S}(n, k)$ whose union satisfies $\bigcup_{S \in C} S = [n]$. Note that by definition, the subsets in C must be pairwise disjoint.

Given some subset $S = \{j_1, \dots, j_\ell\} \subseteq [n]$, let X_S denote the joint random variable $(X_{j_1}, \dots, X_{j_\ell})$. Fix some cluster $C = \{S_1, \dots, S_d\}$ of $\mathcal{P}$. The chain rule of conditional entropy implies that

$$\begin{aligned} H\left(X_1, \dots, X_n \mid Y\right) &= H\left(X_{S_1} \mid Y\right) + H\left(X_{S_2} \mid X_{S_1} \mid Y\right) + \cdots + H\left(X_{S_d} \mid X_{S_1 \cup \cdots \cup S_{d-1}} \mid Y\right) \\ &\leq H\left(X_{S_1} \mid Y\right) + H\left(X_{S_2} \mid Y\right) + \cdots + H\left(X_{S_d} \mid Y\right) . \end{aligned}$$

Denoting the clusters of $\mathcal{P}$ by $C^1, \dots, C^{\binom{n}{k}/d}$ and letting $C^i = \{S^i_1, \dots, S^i_d\}$ for $i = 1, \dots, \binom{n}{k}/d$, we can sum over all clusters of $\mathcal{P}$ to conclude that

$$\frac{\binom{n}{k}}{d} H\left(X_1, \dots, X_n \mid Y\right) \leq \sum_{i=1}^{\binom{n}{k}/d} \sum_{j=1}^{d} H\left(X_{S^i_j} \mid Y\right) . \tag{A-1}$$

The assertion follows since the right hand side of (A-1) has $\binom{n}{k}$ terms, each identified with a unique subset $S \in \mathcal{S}(n, k)$, hence if we pick one term u.a.r., then its expected value is at least $H(X_1, \dots, X_n \mid Y)/d$.

Now, assume that $n = k \cdot d - r$ for some integers $d \geq 1$ and $0 < r < k$ and let $n' = k \cdot d$. Let $X_{n+1}, \dots, X_{n'}$ be r *dummy* random variables with 0 entropy. We have already shown that if subset $S \subseteq [n']$ is chosen u.a.r. from $\mathcal{S}(n', k)$, then

$$d \cdot \mathbb{E}_S\left[H\left(X_S \mid Y\right)\right] \geq H\left(X_1, \dots, X_{n'} \mid Y\right) = H\left(X_1, \dots, X_n \mid Y\right) .$$

Since $H\left(X_S \mid Y\right) = H\left(X_{S \cap [n]} \mid Y\right)$ for every $S \in \mathcal{S}(n', k)$, it follows that shifting the probability mass in a uniform manner from subsets S containing dummy variables to subsets S that do not contain dummy variables cannot decrease the expected entropy. In other words, if subset $S \subseteq [n]$ is chosen u.a.r. from $\mathcal{S}(n, k)$ and subset $S' \subseteq [n']$ is chosen u.a.r. from $\mathcal{S}(n', k)$, then

$$\mathbb{E}_S\left[H\left(X_S \mid Y\right)\right] \geq \mathbb{E}_{S'}\left[H\left(X_S \mid Y\right)\right] .$$

The assertion follows since $d = \lceil n/k \rceil$.

REFERENCES

K.J. Ahn and S. Guha. 2009. Graph Sparsification in the Semi-streaming Model. In *ICALP*. 328–338.

Noga Alon, Baruch Awerbuch, Yossi Azar, Niv Buchbinder, and Joseph Naor. 2009. The Online Set Cover Problem. *SIAM J. Comput.* 39, 2 (2009), 361–370.

Noga Alon, Yuval Emek, Michal Feldman, and Moshe Tennenholtz. 2013. Adversarial Leakage in Games. *SIAM J. Discrete Math.* 27, 1 (2013), 363–385.

Sepehr Assadi, Sanjeev Khanna, and Yang Li. 2016. Tight Bounds for Single-Pass Streaming Complexity of the Set Cover Problem. *CoRR* abs/1603.05715 (2016). http://arxiv.org/abs/1603.05715

Surender Baswana. 2008. Streaming Algorithm for Graph Spanners—single Pass and Constant Processing Time Per Edge. *Inf. Process. Lett.* 106, 3 (April 2008), 110–114.

Amit Chakrabarti and Anthony Wirth. 2016. Incidence Geometries and the Pass Complexity of Semi-Streaming Set Cover. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, Robert Krauthgamer (Ed.). SIAM, 1365–1373. DOI:http://dx.doi.org/10.1137/1.9781611974331.ch94

Erik D. Demaine, Piotr Indyk, Sepideh Mahabadi, and Ali Vakilian. 2014. On Streaming and Communication Complexity of the Set Cover Problem. In *Distributed Computing - 28th International Symposium, DISC 2014, Austin, TX, USA, October 12-15, 2014. Proceedings (Lecture Notes in Computer Science)*, Fabian Kuhn (Ed.), Vol. 8784. Springer, 484–498. DOI:http://dx.doi.org/10.1007/978-3-662-45174-8_33

Michael Elkin. 2011. Streaming and Fully Dynamic Centralized Algorithms for Constructing and Maintaining Sparse Spanners. *ACM Trans. Algorithms* 7, 2 (March 2011), 20:1–20:17.

Yuval Emek, Magnús M. Halldórsson, and Adi Rosén. 2012. Space-Constrained Interval Selection. In *ICALP (1)*. 302–313.

Yuval Emek and Adi Rosén. 2014. Semi-Streaming Set Cover (Extended Abstract). In *ICALP (1)*. 453–464.

Leah Epstein, Asaf Levin, Julián Mestre, and Danny Segev. 2011. Improved Approximation Guarantees for Weighted Matching in the Semi-streaming Model. *SIAM J. Discrete Math.* 25, 3 (2011), 1251–1265. DOI:http://dx.doi.org/10.1137/100801901

Hossein Esfandiari, Mohammad Taghi Hajiaghayi, Vahid Liaghat, Morteza Monemizadeh, and Krzysztof Onak. 2015. Streaming Algorithms for Estimating the Matching Size in Planar Graphs and Beyond, See Indyk [2015], 1217–1233. DOI:http://dx.doi.org/10.1137/1.9781611973730.81

Joan Feigenbaum, Sampath Kannan, Andrew McGregor, Siddharth Suri, and Jian Zhang. 2005. On graph problems in a semi-streaming model. *Theor. Comput. Sci.* 348, 2 (2005), 207–216.

Joan Feigenbaum, Sampath Kannan, Andrew McGregor, Siddharth Suri, and Jian Zhang. 2008. Graph Distances in the Data-Stream Model. *SIAM J. Comput.* 38, 5 (2008), 1709–1727.

Pierre Fraigniaud, Magnús M. Halldórsson, Boaz Patt-Shamir, Dror Rawitz, and Adi Rosén. 2016. Shrinking Maxima, Decreasing Costs: New Online Packing and Covering Problems. *Algorithmica* 74, 4 (2016), 1205–1223. DOI:http://dx.doi.org/10.1007/s00453-015-9995-8

Fabrizio Grandoni, Anupam Gupta, Stefano Leonardi, Pauli Miettinen, Piotr Sankowski, and Mohit Singh. 2013. Set Covering with Our Eyes Closed. *SIAM J. Comput.* 42, 3 (2013), 808–830.

Bjarni V. Halldórsson, Magnús M. Halldórsson, Elena Losievskaja, and Mario Szegedy. 2010. Streaming Algorithms for Independent Sets. In *ICALP*. 641–652.

Piotr Indyk (Ed.). 2015. *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*. SIAM. DOI:http://dx.doi.org/10.1137/1.9781611973730

Piotr Indyk, Sepideh Mahabadi, and Ali Vakilian. 2015. Towards Tight Bounds for the Streaming Set Cover Problem. *CoRR* abs/1509.00118 (2015). http://arxiv.org/abs/1509.00118

Lujun Jia, Guolong Lin, Guevara Noubir, Rajmohan Rajaraman, and Ravi Sundaram. 2005. Universal Approximations for TSP, Steiner Tree, and Set Cover. In *STOC*. 386–395.

David S. Johnson. 1974. Approximation algorithms for combinatorial problems. *J. Comput. System Sci.* 9, 3 (1974), 256 – 278.

Michael Kapralov, Sanjeev Khanna, and Madhu Sudan. 2015. Streaming Lower Bounds for Approximating MAX-CUT, See Indyk [2015], 1263–1282. DOI:http://dx.doi.org/10.1137/1.9781611973730.84

Michael Kapralov, Yin Tat Lee, Cameron Musco, Christopher Musco, and Aaron Sidford. 2014. Single Pass Spectral Sparsification in Dynamic Streams. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*. IEEE Computer Society, 561–570. DOI:http://dx.doi.org/10.1109/FOCS.2014.66

Richard M. Karp. 1972. Reducibility Among Combinatorial Problems. In *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher (Eds.). Plenum Press, 85–103.

Jonathan A. Kelner and Alex Levin. 2013. Spectral Sparsification in the Semi-streaming Setting. *Theory Comput. Syst.* 53, 2 (2013), 243–262.

C. Konrad, F. Magniez, and C. Mathieu. 2012. Maximum Matching in Semi-Streaming with Few Passes. In *APPROX*. 231–242.

Ilan Kremer, Noam Nisan, and Dana Ron. 1999. On Randomized One-Round Communication Complexity. *Computational Complexity* 8, 1 (1999), 21–49.

Charles C. Lindner and Christopher A. Rodger. 2011. *Design Theory* (2nd ed.). CRC Press.

Andrew McGregor. 2005. Finding Graph Matchings in Data Streams. In *APPROX-RANDOM*. 170–181.

Andrew McGregor. 2014. Graph stream algorithms: a survey. *SIGMOD Record* 43, 1 (2014), 9–20. DOI:http://dx.doi.org/10.1145/2627692.2627694

Michael Mitzenmacher and Eli Upfal. 2005. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press. http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20\&path=ASIN/0521835402

Barna Saha and Lise Getoor. 2009. On Maximum Coverage in the Streaming Model & Application to Multi-topic Blog-Watch. In *Proceedings of the SIAM International Conference on Data Mining, SDM 2009, April 30 - May 2, 2009, Sparks, Nevada, USA*. SIAM, 697–708. DOI:http://dx.doi.org/10.1137/1.9781611972795.60

J. H. van Lint and R. M. Wilson. 2001. *A Course in Combinatorics* (2nd ed.). Cambridge University Press.

Vijay V. Vazirani. 2001. *Approximation algorithms*. Springer-Verlag New York, Inc., New York, NY, USA.