# Optimal Bounds for Dominating Set in Graph Streams

**Sanjeev Khanna**
Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, US

**Christian Konrad**
Department of Computer Science, University of Bristol, UK

## Abstract

We resolve the space complexity of one-pass streaming algorithms for Minimum Dominating Set (MDS) in both insertion-only and insertion-deletion streams (up to poly-logarithmic factors) where an input graph is revealed by a sequence of edge updates. Recently, streaming algorithms for the related Set Cover problem have received significant attention. Even though MDS can be viewed as a special case of Set Cover, it is however harder to solve in the streaming setting since the input stream consists of individual edges rather than entire vertex-neighborhoods, as is the case in Set Cover.

We prove the following results ($n$ is the number of vertices of the input graph):

1. In insertion-only streams, we give a one-pass semi-streaming algorithm (meaning $\tilde{\mathrm{O}}(n)$ space) with approximation factor $\tilde{\mathrm{O}}(\sqrt{n})$. We also prove that every one-pass streaming algorithm with space $o(n)$ has an approximation factor of $\Omega(n/\log n)$.
   Combined with a result by [Assadi et al., STOC'16] for Set Cover which, translated to MDS, shows that space $\tilde{\Theta}(n^2/\alpha)$ is necessary and sufficient for computing an $\alpha$-approximation for every $\alpha = o(\sqrt{n})$, this completely settles the space requirements for MDS in the insertion-only setting.
2. In insertion-deletion streams, we prove that space $\Omega(n^2/(\alpha \log n))$ is necessary for every approximation factor $\alpha \leq \Theta(n/\log^3 n)$. Combined with the Set Cover algorithm of [Assadi et al., STOC'16], which can be adapted to MDS even in the insertion-deletion setting to give an $\alpha$-approximation in $\tilde{\mathrm{O}}(n^2/\alpha)$ space, this completely settles the space requirements for MDS in the insertion-deletion setting.

**2012 ACM Subject Classification** Theory of computation → Streaming models; Theory of computation → Lower bounds and information complexity; Theory of computation → Graph algorithms analysis

**Keywords and phrases** Streaming algorithms, communication complexity, information complexity, dominating set

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2022.93

**Funding** *Sanjeev Khanna*: Supported in part by NSF awards CCF-1910534, CCF-1926872, and CCF-2045128.
*Christian Konrad*: Supported by EPSRC New Investigator Award EP/V010611/1.

## 1 Introduction

Streaming algorithms for processing large graphs have been studied since more than two decades [17]. In the most traditional setting, the one-pass *insertion-only* setting, an algorithm receives the input graph $G = (V, E)$ with $|V| = n$ as a sequence of its edges in arbitrary order. The algorithm is required to maintain a memory of size sublinear in the input size and to output a solution once the entire stream has been processed. A natural and well-studied extension is the *insertion-deletion* setting, where previously inserted edges can be deleted again. In both streaming models, the aim is to determine the space necessary and sufficient for algorithms to solve a specific task.

13th Innovations in Theoretical Computer Science Conference (ITCS 2022).
Editor: Mark Braverman; Article No. 93; pp. 93:1–93:23
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

In this paper, we initiate the study of the Minimum Dominating Set (MDS) problem in the one-pass streaming model and resolve its space complexity in both the insertion-only and insertion-deletion settings. A *dominating set* $D \subseteq V$ in a graph $G = (V, E)$ is a subset of vertices that dominate or *cover* all other vertices, i.e., every vertex in $V \setminus D$ is adjacent to at least one vertex in $D$. A *minimum dominating set* is one of smallest size, and the size of a minimum dominating set in a graph $G$ is known as its *domination number*, denoted $\gamma(G)$. The objective of the MDS problem is to compute a minimum dominating set or an approximation thereof:

▶ **Definition 1** ($\alpha$-approximation to MDS). *A (streaming) algorithm for MDS is an $\alpha$-approximation algorithm if, on every input graph $G = (V, E)$, the algorithm outputs a dominating set $D \subseteq V$ of size at most $\alpha \cdot \gamma(G)$ and a cover certificate $C : V \rightarrow D$, indicating for every vertex $v \in V$ a vertex $C(v) \in D$ that* covers *$v$, i.e., that is contained in the inclusive neighborhood of $v$.*

MDS is closely related to the Set Cover problem, which has recently received significant attention in the streaming setting [13, 9, 14, 5, 16, 18, 3, 7]. In Set Cover, the input consists of a universe $\mathcal{U}$ of size $|\mathcal{U}| = n$ and a collection of $m$ sets $\mathcal{S} = \{S_1, \dots, S_m\}$ with $S_i \subseteq \mathcal{U}$, for every $i$. The output is a smallest subset $\mathcal{S}' \subseteq \mathcal{S}$ that *covers* $\mathcal{U}$, i.e., such that $\cup_{S \in \mathcal{S}'} S = \mathcal{U}$. Similar to MDS, an $\alpha$-approximate set cover is one of size at most $\alpha$ times the size of a smallest set cover.

In a non-streaming context, MDS and Set Cover are essentially equivalent since the two problems can be reduced to each other in linear time. For example, in order to solve MDS on a graph $G = (V, E)$ with $V = \{v_1 \dots v_n\}$ using an algorithm for Set Cover, we create a Set Cover instance with $n$ input sets $\mathcal{S} = \{S_1, \dots, S_n\}$ where, for every $1 \leq i \leq n$, $S_i$ is the inclusive neighborhood of $v_i$, i.e., $S_i = \Gamma[v_i]$. A set cover in this instance then immediately yields a dominating set in $G$ of the same size. In the streaming setting, however, the natural ways in which the inputs are presented in the respective input streams are different, and the two problems therefore cannot directly be reduced to each other. In Set Cover, all previous work (with the exception of [18], see Subsection 1.3, further related work) assumes that entire sets arrive one-by-one in the input stream. In MDS, it is more natural to consider the aforementioned insertion-only graph stream setting, where edges arrive one-by-one in the stream. We observe that, from an algorithmic perspective, the edge-arrival setting for MDS can only be harder since none of the inclusive neighborhoods are visible in the stream at any one moment. Conversely, lower bounds for streaming Set Cover on instances with $\Theta(n)$ sets carry over to MDS.

Set Cover in the one-pass streaming model is well-understood and exhibits an interesting *phase-transition* when the desired approximation factor is $\alpha = \Theta(\sqrt{n})$. There are simple semi-streaming algorithms (i.e., with $\tilde{\mathrm{O}}(n)$ space) that achieve an approximation factor of $O(\sqrt{n})$ [14, 9], regardless of the number of sets. But for any approximation factor $\alpha = o(\sqrt{n})$, it is known that space $\tilde{\Theta}(\frac{mn}{\alpha})$ is necessary and sufficient [5]. In other words, while $\tilde{\mathrm{O}}(n)$ space is enough for an $O(\sqrt{n})$-approximation, any algorithm that is tasked with achieving a slightly better approximation factor of $o(\sqrt{n})$ necessarily requires $\Omega(m\sqrt{n})$ space. A natural question is if MDS also exhibits a similar phase transition at some approximability threshold? And if so, does it occur at a much higher approximability threshold given that in MDS, elements in each set arrive separately arbitrarily interspersed with elements of other sets? Finally, for many graph problems, including Connectivity [1], Cut-sparsifiers and Spectral-sparsifiers [1, 2, 20, 21], and $(\Delta + 1)$-coloring [4], it is known that the space-approximation tradeoffs known for insertion-only streams can be extended to insertion-deletion streams at the expense of only a poly-logarithmic factor larger space. However, notable exceptions to this are the

Maximum Matching and the Minimum Vertex Cover problems where the space needed for insertion-deletion streams can be shown to be $\Omega(n)$ factor larger than the space requirements for insertion-only setting [22, 6, 12]. Does the space-approximation tradeoff for MDS also qualitatively change as we go from insertion-only streams to the more general model of insertion-deletion streams?

In this work, we answer all the above-mentioned questions, obtaining tight space-approximation tradeoffs for MDS in both insertion-only and in insertion-deletion streams.

## 1.1 Our Results

Our first main result is that there is an $\tilde{\mathrm{O}}(n)$ space algorithm for MDS that achieves $\tilde{\mathrm{O}}(\sqrt{n})$-approximation, showing that the semi-streaming approximability threshold for MDS is essentially the same as that for Set Cover.

▶ **Theorem 2.** *There is a randomized one-pass semi-streaming algorithm for MDS that achieves $\tilde{\mathrm{O}}(\sqrt{n})$-approximation.*

Our algorithm can in fact solve Set-Cover instances that are presented in the edge-arrival model, i.e., as an arbitrary sequence of tuples $(u, S_i) \in \mathcal{U} \times \mathcal{S}$, indicating that item $u$ is contained in set $S_i$, with space $\tilde{\mathrm{O}}(|\mathcal{U}| + |\mathcal{S}|)$ space. Observe that instances with $|\mathcal{S}|$ sets may require space $\tilde{\mathrm{O}}(|\mathcal{U}| \cdot |\mathcal{S}|)$ to write down – our algorithm therefore uses up to a factor of $n = |\mathcal{U}|$ less memory than the input instance size. The theorem below summarizes this result.

▶ **Theorem 3.** *There is a randomized one-pass streaming algorithm for Set-Cover in the edge-arrival model that achieves $\tilde{\mathrm{O}}(\sqrt{n})$-approximation using $\tilde{\mathrm{O}}(n + m)$ space where $n = |\mathcal{U}|$, and $m = |\mathcal{S}|$.*

Once we consider the $o(\sqrt{n})$-approximation regime, the space-approximation tradeoffs for MDS can be easily shown to converge to that for Set Cover instances with $n$ sets. On the one hand, the one-pass $\tilde{\Theta}(\frac{mn}{\alpha})$ space streaming algorithm for Set Cover in [5], which we call the AKL-algorithm (Assadi-Khanna-Li [5]) and expand on in Appendix A, can easily be applied to MDS with parameter $m = n$ for all values of $\alpha$, and even works in the insertion-deletion setting with only a poly-logarithmic increase in the space requirements. On the other hand, [5] shows a matching $\Omega(\frac{mn}{\alpha})$ space lower bound for insertion-only streams that holds for $\alpha = o(\sqrt{n})$, and also carries over to MDS with parameter $m = \Theta(n)$. These two result combined show that $\tilde{\Theta}(n^2/\alpha)$ space is necessary and sufficient for MDS in both the insertion-only and the insertion-deletion settings, for every $\alpha = o(\sqrt{n})$. Thus in the setting of insertion-only streams, MDS behaves in a similar manner to Set Cover, exhibiting a phase transition at $\alpha = \tilde{\mathrm{O}}(\sqrt{n})$.

We further complete the picture for insertion-only streams by observing that if one tries to go below the regime of semi-streaming space then not much more than a trivial $O(n)$-approximation factor is achievable.

▶ **Theorem 4.** *Any constant error one-pass streaming algorithm for MDS in the insertion-only model with approximation factor $o(n/\log n)$ requires $\Omega(n)$ space.*

Our second main result is that the landscape of space-approximation trade-offs fundamentally changes when we consider MDS in insertion-deletion streams. In particular, in the semi-streaming space regime, no algorithm can achieve much better than the trivial $O(n)$-approximation.

▶ **Theorem 5.** *Every insertion-deletion streaming algorithm for MDS with approximation factor $\alpha$, for any $\alpha \leq \Theta(n/\log^3 n)$, requires space $\tilde{\mathrm{O}}(\frac{n^2}{\alpha})$.*

**Table 1** Space/approximation trade-offs for MDS in the one-pass streaming model.

**(a)** Insertion-only model.

| Space | Approx. | Reference |
|---|---|---|
| $o(n)$ | $\tilde{\Omega}(n)$ | Theorem 4 |
| $\tilde{\Theta}(n)$ | $\tilde{O}(\sqrt{n})$ | Theorem 2 |
| $\tilde{\Theta}(n^2/\alpha)$ | $\alpha = o(\sqrt{n})$ | Assadi et al. [5] |

**(b)** Insertion-deletion model.

| Space | Approx. | Reference |
|---|---|---|
| $\tilde{\Omega}(n^2/\alpha)$ | $\alpha \leq \Theta(\frac{n}{\log^3 n})$ | Theorem 5 |
| $\tilde{O}(n^2/\alpha)$ | $\alpha$ | Assadi et al. [5] |

Our results, together with Assadi et al.'s results for Set Cover [5] applied to MDS, completely characterize the space/approximation trade-offs for MDS in both the insertion-only and the insertion-deletion models, see Table 1.

## 1.2 Techniques

We now give an overview of the technical ideas underlying our results. We start with an overview of the algorithm underlying Theorem 2.

Let $H = (V, F)$ be the $n$-vertex input graph. Instead of working with $H$ directly, we consider the *bipartite incidence graph* $G = (A, B, E)$ with $A = B = V$ instead: For every edge $\{u, v\} \in F$, we add the edges $(u, v)$ and $(v, u)$ to $G$, using the notation $(a, b)$ to denote an edge in the bipartite incidence graph where $a \in A$ and $b \in B$. When working with $G$, the objective is to select a subset $D \subseteq A$ such that $\Gamma_G(D) = B$ and $|D|$ is minimum.

To motivate our algorithm, it will be instructive to first consider the simplified setting where each vertex in $A$ arrives with all its incident edges to vertices in $B$ (as in the standard Set Cover problem). In that case, we can simply maintain a set $S \subseteq B$ of currently undominated vertices, and whenever a vertex $a \in A$ arrives that can dominate at least $\sqrt{n}$ vertices in $S$, we include it in our solution and remove this vertex along with its neighbors from $S$. Let $S'$ denote the set $S$ upon termination of the algorithm – we include all vertices in $S'$ in our dominating set solution. This algorithm can clearly be implemented in $\tilde{O}(n)$ space. It is also easy to see that the algorithm chooses at most $\sqrt{n}$ vertices in processing the stream as inclusion of any vertex in the solutions removes at least $\sqrt{n}$ vertices from $S$. Moreover, no vertex in any optimal solution could dominate more than $\sqrt{n}$ vertices in $S'$ (or else we would have included it in our solution). Together, these two observations imply that this is an $O(\sqrt{n})$-approximation semi-streaming algorithm.

However, in the setting of MDS, we only see edges arrive in some arbitrary manner. The challenge then is that using only $\tilde{O}(1)$ amount of information per vertex we need to recognize (i) when inclusion of a vertex in our solution can allow us to dominate many vertices, and (ii) for each dominated vertex, record the vertex in our solution that dominates it. On the surface, the first task can be accomplished by just maintaining the degree of each vertex to currently undominated vertices. The difficulty here is that the degree information alone does not allow us to differentiate between the scenario where the high degree vertices in $A$ dominate different subsets of vertices in $B$ and the scenario when they dominate a common set of vertices in $B$. In the latter scenario, we should not be including all the high-degree vertices in our solution. We handle this by setting up a probabilistic process which includes vertices with high degree with a suitable probability that avoids taking too many vertices in one go. Vertices with continued high-degree get multiple chances for inclusion with geometrically increasing probability of inclusion. But how do we recognize which vertices have been covered by vertices chosen in our solution? We accomplish this by decoupling the selection process from the covering process. In particular, we start recording which vertices are covered by a vertex $v$ selected in our solution only after the vertex $v$ has been selected in our solution. Intuitively, if a vertex is going to cover many vertices, then the probabilistic selection process ensures that many edges incident on this vertex will arrive after it has been selected. Together, this ensures that using only $\tilde{\mathrm{O}}(1)$ amount of information per vertex, the algorithm is both $\tilde{\mathrm{O}}(\sqrt{n})$-competitive with the optimal solution, and is able to maintain a certificate of coverage for each vertex.

We next describe the ideas underlying proofs of our lower bound results in Theorems 4 and 5. As is typically the case for streaming lower bounds, our lower bounds are proved in the two-party communication setting, where the edge set of the input graph is partitioned between Alice and Bob. In the insertion-deletion setting, in addition to his share of input edges, Bob also holds edge deletions, which form a subset of Alice's input edges. It will be helpful to first consider the lower bound for the insertion-only setting (Theorem 4), where we prove that every one-pass streaming algorithm with space $o(n)$ has an approximation factor of $\Omega(n/\log n)$, and then build on this construction to illustrate the much more involved $\Omega(n^2/\alpha)$ space lower bound for $\alpha$-approximation algorithms in the insertion-deletion setting (Theorem 5).

Consider the following input graph $G = (\{v_A, v_B\} \cup [n], E)$ on $n + 2$ vertices, where every edge is incident to either $v_A$ or $v_B$. Alice holds the edges incident to $v_A$ and Bob holds the edges incident to $v_B$. Let $\Gamma_G(v)$ denote the set of vertices adjacent to a vertex $v$, and let $\Gamma_G[v] = \Gamma_G(v) \cup \{v\}$. Suppose that these edges are such that $\Gamma_G[v_A] \cup \Gamma_G[v_B] = ([n] \cup \{v_A, v_B\}) \setminus \{T\}$ with $T \in [n]$, i.e., vertices $v_A$ and $v_B$ cover all but the single vertex $T$. The set $\{v_A, v_B, T\}$ thus constitutes a dominating set of size 3, and any dominating set $D$ that constitutes an $o(n)$-approximate solution therefore needs to be of size $3 \cdot o(n) = o(n)$. Moreover, the dominating set $D$ is required to contain vertex $T$, since $T$ cannot be covered by $v_A$ or $v_B$. However, the task of identifying $T$ is hard – it is easy to obtain an $\Omega(n)$ lower bound on the two-party communication complexity for this task. On the surface, this appears to immediately yield the desired lower bound for our problem. This argument, however, is incomplete since the output dominating set $D$ does not allow us to identify $T$ *exactly*. In particular, since $|D| = o(n)$ and $T \in D$, we are only able to reduce the possibilities as to the identity of the element $T$ from an initial set of $n$ possibilities to a set of $o(n)$ possibilities. We will, however, see that *approximately* identifying $T$ (for the right notion of approximation) is essentially as hard (up to constants) as identifying $T$ exactly, which establishes our desired lower bound. The final step in the argument above follows by utilizing a result of Assadi et al. [4] on the Set-Union problem where Alice holds a set $A \subseteq [n]$, Bob holds a set $B \subseteq [n]$, and Alice and Bob are guaranteed that $A \cup B = [n] \setminus \{T\}$, for some $T \in [n]$. Assadi et al. proved that any protocol that sufficiently skews the a priori uniform distribution of $T$ substantially has communication cost $\Omega(n)$.

Our lower bound in the insertion-deletion setting is also based on Set-Union, however, the ability to incorporate edge deletions into the construction allows us to hide the Set-Union instance within a larger problem with distinctly higher communication complexity that we refer to as Embedded-Set-Union. In this problem, Alice holds $n$ subsets $A_1, \dots, A_n \subseteq [n]$, Bob holds an index $I \in [n]$ and set $B$ such that $(A_I, B)$ is a Set-Union instance. In addition to $I$ and $B$, Bob also knows all but $\beta = \Theta(n/(\alpha \log n))$ elements of the sets $A_i$, for every $i \neq I$. The objective is to solve the underlying Set-Union instance $(A_I, B)$, i.e., approximately identifying the element $T = [n] \setminus (A_I \cup B)$. As our main lower bound result of this paper, we prove that ESU has communication complexity $\Omega(n\beta)$.

Bob's knowledge about the sets $(A_i)_{i \neq I}$ is pivotal in our construction. When reducing ESU to MDS, we create a graph $G$ with vertex neighborhoods $A_I, B$ and $(A_i)'_{i \neq I}$, where $A'_i \subseteq A_i$ is the subset of elements that is not known to Bob, or, in other words, Bob's knowledge of the sets $A_i$ corresponds to edge deletions in the reduction. These deletions ensure that the surviving sets $(A'_i)_{i \neq I}$ are small enough (of size $\beta$) so as to cover only few vertices of $G$ if they are selected into the output dominating set. Observe that $\gamma(G) = 3$ (via the sets $A_I, B$ and any set that covers $T$). Similar to the insertion-only setting, the hardness stems from the fact that the output dominating set $D$ is required to cover $T$. Since this can only be achieved via the sets $(A'_i)_{i \neq I}$, and only at most $3\alpha$ of these sets can be chosen into $D$ since $D$ is an $\alpha$-approximation, the dominating set algorithm is therefore able to (approximately) identify $T$ as one of the $o(n)$ elements covered by the selected sets from $(A'_i)_{i \neq I}$. This substantially reduces the possible values for $T$ from $n$ to $o(n)$ and thus solves ESU.

Proving a lower bound on the communication complexity of ESU is the most technical contribution of this paper. To this end, we adopt the information complexity framework and measure the amount of information necessary revealed about Alice's input $A_1, \dots, A_n$ in so-called *nice protocols* for ESU, i.e., protocol consisting of only two messages; one from Alice to Bob, and a *relatively short* one from Bob back to Alice that consists of candidate values for $T$, a constraint that we need to impose for technical reasons. We first argue that any nice protocol for ESU can also be used to solve Set-Union, which establishes that any transcript of a nice protocol for ESU necessarily reveals $\Omega(n)$ bits of information about $A_I$, as implied by the lower bound for Set-Union. Recall that, for every $i \neq I$, Bob already knows all but $\beta$ bits of $A_i$. Our aim, therefore, is to show that every nice protocol reveals $\Omega(\beta)$ bits of information about the part of $A_i$ $(i \neq I)$ unknown to Bob, which then yields the desired lower bound by summing up over every $i \neq I$. We establish this by a combination of two arguments. First, using the fact that any nice protocol reveals $\Omega(n)$ bits of information about $A_I$, we prove that any such protocol must also reveal $\Omega(\beta)$ bits of information about a randomly chosen subset $A'_I \subseteq A_I$ of size $\beta$. Second, since Alice does not know the index $I$, we establish that a randomly chosen subset in any other set $A_i$, $i \neq I$, which we ensure to coincide with the bits that Bob does not know by appropriately defining the hard input distribution, also needs to reveal this amount of information, thereby completing the argument.

## 1.3 Further Related Work

We will now expand on further results on the streaming Set-Cover problem. Set-Cover has been extensively studied in the *semi-streaming* model [15], where streaming algorithms are allowed to use $O(n \operatorname{poly} \log n)$ space. Saha and Getoor [26] initiated the study of streaming algorithms for Set-Cover and gave an $O(\log n)$-approximation semi-streaming algorithm that makes $O(\log n)$ passes over the stream. Emek and Rosén [14] were the first to conduct a thorough study of the one-pass semi-streaming setting and showed that an approximation factor of $\Theta(\sqrt{n})$ can be achieved (even in a weighted version of the problem) and that this is best possible. Emek and Rosén's algorithm proved extremely efficient in practice. In a recent study [7], their algorithm achieved cover sizes that are only 8% larger than those produced by a non-streaming disk-friendly algorithm [10] while using between 10-73 times less memory. Chakrabarti and Wirth [9] then extended Emek and Rosén's results to multiple passes and showed that an approximation factor of $\Theta(n^{1/(p+1)})$ is achievable and optimal when $p$ passes over the stream are allowed, for any constant $p$.

Further works that consider algorithms with substantially more space than semi-streaming space are known. Assadi [3] showed that $O(\operatorname{poly} \log n)$ passes $\alpha$-approximation algorithms require space $\Omega(mn^{\frac{1}{\alpha}})$, even if the input stream is in random order. This lower bound is matched by Har-Peled et al.'s $\alpha$-approximation algorithm [16], which uses space $\tilde{O}(mn^{\frac{1}{\alpha}})$ and performs $O(\alpha)$ passes over the input, which can be in adversarial order (see also the earlier work [13] for an algorithm that uses slightly more space).

Last, a fractional version of the Set Cover problem has also been studied in the streaming model. In [18], multi-pass algorithms for this problem based on the multiplicative weights update method are presented. Interestingly, their algorithm also naturally works in an edge-arrival setting similar to the edge streaming model consider in this paper. We stress, however, that their algorithm does not give any results if only few passes are considered, and, as such, their techniques cannot give non-trivial bounds in the one-pass setting.

### 1.4 Organization

The rest of this paper is organized as follows. In Section 2, we give a brief overview of the known tools from communication and information complexity that will be useful in our lower bound proofs. Then in Section 3, we present our semi-streaming algorithm for MDS, proving Theorem 2. We present our lower bound results for MDS, namely, proofs of Theorems 4 and 5 in Section 4. Finally, we conclude with a direction for future work in Section 5.

## 2 Preliminaries

### 2.1 Communication Complexity

We will now provide the necessary context on communication complexity for proving our lower bounds (see the excellent monographs [23, 25, 24] for an overview).

In the *two-party communication complexity framework*, there are two parties, denoted Alice and Bob, who each hold a part of their joint input $(A, B)$. The objective of Alice and Bob is to solve a joint problem by communicating as few bits as possible to each other. The way Alice and Bob interact is specified by a communication protocol $\Pi$. Alice and Bob may use randomization, in which case Alice and Bob have access to a (public) infinite shared string of random bits $R_\Pi$, and also each have access to infinite (private) strings of random bits. With slight abuse of notation, we denote the *transcript* of their communication, i.e., the entirety of all their exchanged messages, also by $\Pi$. The *cost* of a communication protocol $\Pi$, denoted $|\Pi|$, is the maximum total number of bits exchanged in an execution of $\Pi$. Then, the *randomized $\epsilon$-error communication complexity* of problem $\mathsf{P}$, denoted $R_\epsilon(\mathsf{P})$, is the minimum cost of an $\epsilon$-error communication protocol for $\mathsf{P}$.

### 2.2 Inequalities Involving Entropy and Mutual Information

We will use information theory to give lower bounds on the communication complexity of communication problems. To this end, in this section we give notation and useful inequalities involving entropy and mutual information. For more details on information theory, we refer the reader to the monograph by Cover and Thomas [11].

Let $A, B, C$ be jointly distributed random variables according to distribution $\mathcal{D}$. We denote by $H_\mathcal{D}(A)$ the entropy of $A$, and by $H_\mathcal{D}(A \mid B)$ the conditional entropy of $A$ conditioned on $B$. The mutual information between $A$ and $B$ is denoted by $I_\mathcal{D}(A : B)$, and the conditional mutual information of $A$ and $B$ conditioned on $C$ is denoted by $I_\mathcal{D}(A : B \mid C)$. If the distribution $\mathcal{D}$ is clear from the context then we may drop the subscript $\mathcal{D}$ in the entropy and mutual information expressions.

We will make use of the following inequalities: (let $A, B, C, D$ be jointly distributed random variables)

1. If $A$ and $B$ are independent conditioned on $CD$ then $I(A : B \mid CD) = 0$ (see Sec. 2 in [11]).

2. If $A$ and $C$ are independent conditioned on $D$ then $I(A : B \mid CD) \geq I(A : B \mid D)$. (see Prop. B.3. in [4])
3. If $D$ is independent of $A, B$ and $C$ then $I(A : B \mid CD) = I(A : B \mid C)$ (see Sec. 2 in [11]).
4. $I(A : B \mid C) \leq \min\{H(A), H(B)\}$ (immediate from def. of mutual information).

## 2.3 Information Complexity

In order to prove lower bounds on our communication problems, we use the *information complexity* framework (see [8] for a great overview). Information complexity approaches to proving lower bounds on the communication cost of protocols measure the amount of information necessarily revealed by the transcript of the protocol. This quantity is a natural lower bound on the communication cost of the protocol, since the amount of information revealed cannot be larger than the number of bits exchanged.

▶ **Definition 6** (Internal and external information cost). *Denote by $\mathcal{D}_{\mathsf{P}}$ a distribution over inputs $(X, Y)$ for a two-party communication problem* P*, and let $\Pi$ be a protocol. Then, the* internal information cost *of $\Pi$, denoted $\mathrm{ICost}^{int}_{\mathcal{D}}(\Pi)$, and the* external information cost *of $\Pi$, denoted $\mathrm{ICost}^{ext}_{\mathcal{D}}(\Pi)$, are defined as:*

$$\mathrm{ICost}^{int}_{\mathcal{D}}(\Pi) = I_{\mathcal{D}_{\mathsf{P}}}(X : \Pi \mid YR_{\Pi}) + I_{\mathcal{D}_{\mathsf{P}}}(Y : \Pi \mid XR_{\Pi}) \ , \ \textit{and}$$
$$\mathrm{ICost}^{ext}_{\mathcal{D}}(\Pi) = I_{\mathcal{D}_{\mathsf{P}}}(XY : \Pi \mid R_{\Pi}) \ .$$

*The* internal (external) information complexity *of problem P, denoted $\mathrm{IC}^{int}_{\mathcal{D}}(P)$ (resp. $\mathrm{IC}^{ext}_{\mathcal{D}}(P)$), is the minimum internal (resp. external) information cost of any randomized constant error protocol that solves P.*

It is well-known (e.g. [8]) that the information complexity (both internal and external) of a problem P constitutes a lower bound on the randomized constant-error communication complexity of P. It is therefore enough to bound the information complexity of a problem rather than the communication complexity directly.

Last, we will use the fact that external information cost cannot be smaller than internal information cost:

▶ **Lemma 7** (e.g. [8]). *Let $\Pi$ be a two-party communication protocol. Then:*

$$\mathrm{ICost}^{int}_{\mathcal{D}}(\Pi) \leq \mathrm{ICost}^{ext}_{\mathcal{D}}(\Pi) \ .$$

# 3 Semi-streaming Algorithm in the Insertion-only Model

We now give an algorithm for MDS in the one-pass edge-arrival streaming model. Let $H = (V, F)$ be the $n$-vertex input graph. Recall that instead of working with $H$ directly, we consider the *bipartite incidence graph* $G = (A, B, E)$ with $A = B = V$ instead: for every edge $\{u, v\} \in F$, we add the edges $(u, v)$ and $(v, u)$ to $G$, using the notation $(a, b)$ to denote an edge in the bipartite incidence graph where $a \in A$ and $b \in B$. The objective now is to select a subset $D \subseteq A$ such that $\Gamma_G(D) = B$ and $|D|$ is minimum. In what follows, we present an $\tilde{\mathrm{O}}(\sqrt{n})$-approximation algorithm for MDS using $\tilde{\mathrm{O}}(|A| + |B|) = \tilde{\mathrm{O}}(n)$ space.

Observe that when going from input graph $H$ to the bipartite incidence graph $G$ in the streaming model, for every edge $\{u, v\} \in F$ arriving in the stream, the edges $(u, v)$ and $(v, u)$ are fed into our algorithm. The additional structure, i.e., the knowledge that edge $(v, u)$ follows edge $(u, v)$, is not exploited by Algorithm 1. In other words, Algorithm 1 also works as an $\tilde{O}(\sqrt{n})$-approximation algorithm for a general Set Cover instance with $n$ sets and a universe of size $n$, and, as we will point out further below, can be extended with minimal modification to solve Set Cover instances on $m$ sets with approximation factor $\tilde{O}(\sqrt{n})$ using only $\tilde{O}(m+n)$ space.

## 3.1 The Algorithm

Our algorithm maintains sets $D_0, D_1, ..., D_{\log n}$ that are all initially empty such that each set $D_i$ will contain only $O(\sqrt{n})$ vertices in expectation, and at most $O(\sqrt{n}\log n)$ vertices with high probability. Our dominating set solution will include all vertices in $\cup_{i=0}^{\log n} D_i$, as well as any vertices at the end that are not dominated by vertices in $\cup_{i=0}^{\log n} D_i$. Since $\sum_{i=0}^{\log n} |D_i|$ is $O(\sqrt{n}\log n)$, the goal of the algorithm in choosing vertices for inclusion in $\cup_{i=0}^{\log n} D_i$ is to ensure that the size of the set of undominated vertices that remains at the end is only $O(\sqrt{n}\log n)$ times the size of an optimal solution. Towards this end, we ensure that if there is any vertex in $A$ that could have dominated $\Omega(\sqrt{n})$ undominated vertices at the end, it is necessarily included in $\cup_{i=0}^{\log n} D_i$. Combined together, these properties imply a solution that is only $O(\sqrt{n}\log n)$ as large as the optimal solution in expectation ,and $O(\sqrt{n}\log^2 n)$ as large as the optimal solution with high probability.

As outlined in Section 1.2, the challenge in achieving the above-mentioned properties in the semi-streaming regime is that we only have $\tilde{O}(1)$ space available per vertex to recognize that it can dominate many vertices not yet dominated as well as maintain a certificate of coverage for each dominated vertex. We achieve the former by continually recording degrees to undominated vertices and including a vertex in the set $D_i$ with probability $2^i/\sqrt{n}$ if it has essentially accumulated $\Theta(\sqrt{n})$ degree to undominated vertices for $i$ successive iterations. This probabilistic inclusion process ensures that on the one hand, we do not include too many vertices that are all dominating the same set of vertices, and yet upon termination of the algorithm, any vertices that can dominate $\Omega(\sqrt{n}\log n)$ undominated vertices are necessarily included in $\cup_{i=0}^{\log n} D_i$ – a consequence of the aggressive ramping up of inclusion probabilities. Finally, the algorithm handles the recording of certificate of coverage in small space by starting to record the vertices that will be covered by a vertex $v$ included in our solution only once the vertex $v$ is already included in the solution. The probabilistic inclusion process ensures that many edges incident on an included vertex $v$ are expected to arrive only after $v$ has been selected. Altogether, this ensures that using only $\tilde{O}(1)$ amount of information per vertex, the algorithm is both $\tilde{O}(\sqrt{n})$-competitive with the optimal solution, and is able to maintain a certificate of coverage for each vertex.

## 3.2 Analysis

It is easy to verify that the space used by the algorithm is $\tilde{O}(|A|+|B|) = \tilde{O}(n)$ as we are only maintaining degrees of vertices in $A$, recording which vertices in $B$ have been covered, and storing the subset of $A$ that is included in our solution. In what follows, we show that the expected size of the solution returned by our algorithm is $O(\sqrt{n}\log n)$ times the optimal.

We call $d(a)$ the *uncovered degree* of vertex $a$. For any vertex $b \in B$ and integer $i \geq 0$, we say that an edge $e = (a, b)$ is a *level-$i$* edge if at the moment when $e$ arrived in the stream, $i \cdot n^{1/2} \leq d(a) < (i+1) \cdot n^{1/2}$. Let $X_i(b)$ be a random variable whose value equals the number of level-$i$ edges incident on a vertex $b \in B$, and let $X_i = \sum_{b \in B} X_i(b)$ denote the total number of level-$i$ edges.

**Algorithm 1** Single-pass Semi-Streaming Algorithm for MDS.

```
Require: Bipartite input graph $G = (A, B, E)$ with $|A| = |B| = n$
Let $D_1, D_2, \dots, D_{\log n} \leftarrow \{\}$
For every $a \in A$: $d(a) \leftarrow 0$
$U \leftarrow \varnothing$ {Keep track of dominated nodes ($U \subseteq B$ always holds)}
For every $b \in B : C(b) \leftarrow \perp$ {Output cover certificate}
Let $D_0 \subseteq A$ such that every vertex is included in $D_0$ with probability $p_0 := \frac{1}{\sqrt{n}}$
while stream not empty do
    Let $(a, b)$ be the next edge in the stream
    if $b \in U$ then {ignore edge if incident to already covered $B$-vertex}
        Continue with next edge in stream
    end if
    {vertex $b$ is not yet covered}
    $d(a) \leftarrow d(a) + 1$
    if $d(a) = i \cdot n^{1/2}$ for some integer $i \geq 1$ then
        With probability $p_i := \frac{2^i}{\sqrt{n}} = 2^i p_0$: $D_i \leftarrow D_i \cup \{a\}$
    end if
    if $a \in \bigcup_{i \geq 0} D_i$ then {$b$ is dominated by $a$}
        $U \leftarrow U \cup \{b\}$
        $C(b) \leftarrow a$
    end if
end while
For every $b \in B \setminus U : C(b) \leftarrow b$
return Dominating set $\cup_{i=0}^{\log n} D_i \cup (B \setminus U)$ and cover certificate $C$
```

▶ **Lemma 8.** *For any integer $i \geq 0$, we have:*

$$\mathbb{E}[X_i] \leq \frac{n^{3/2}}{2^i} .$$

**Proof.** We will show that for any vertex $b \in B$, and $i \geq 0$, $\mathbb{E}[X_i(b)] \leq \frac{\sqrt{n}}{2^i}$ . The lemma then follows by linearity of expectation as

$$\mathbb{E}[X_i] = \mathbb{E}\left[\sum_{b \in B} X_i(b)\right] = \sum_{b \in B} \mathbb{E}[X_i(b)] \leq \frac{n^{3/2}}{2^i}.$$

Now to bound $\mathbb{E}[X_i(b)]$, we observe that if an edge $(a, b)$ is a level-$i$ edge then it means that vertex $a$ was sampled with probability $2^i p_0$ (and not included). Let $Z$ be a random variable that denotes the number of trials needed to see a success when each trial has success probability $2^i p_0$. Then $X_i(b) \preceq Z$, that is, $Z$ stochastically dominates $X_i(b)$ since for any positive integer $K$, $\Pr[X_i(b) = K] \leq \Pr[Z = K]$. Note that $X_i(b)$ may be much smaller than $Z$ as the vertex $b$ stops accumulating level-$i$ edges as soon as all edges incident on vertex $b$ have arrived even if the Bernoulli process defined above has not seen a success. Now since the random variable $Z$ is distributed according to geometric distribution, we have

$$\mathbb{E}[X_i(b)] \leq \mathbb{E}[Z] \leq \frac{1}{2^i p_0} = \frac{\sqrt{n}}{2^i}.$$ ◀

▶ **Lemma 9.** *For any integer $i \geq 0$, let $A_i \subseteq A$ be the set of vertices $a$ with $d(a) \geq i \cdot n^{1/2}$ at the end of the stream. Then*

$$\mathbb{E}[|A_i|] \leq \frac{n}{2^{i-1}} \; .$$

**Proof.** For any vertex $a \in A$ to be included in the set $A_i$, the vertex $a$ must receive $\sqrt{n}$ level-$(i-1)$ edges that are incident on it. But by Lemma 8, we know that the expected number of level-$(i-1)$ edges is bounded by $n^{3/2}/2^{i-1}$. Thus $|A_i| \leq X_{i-1}/\sqrt{n}$, and hence $\mathbb{E}[A_i] \leq \mathbb{E}[X_{i-1}]/\sqrt{n} \leq \frac{n}{2^{i-1}}$. ◀

▶ **Lemma 10.** *At the end of the stream, for every integer $i \geq 0$, we have $\mathbb{E}[|D_i|] \leq 2\sqrt{n}$.*

**Proof.** For any $i \geq 0$, the set $D_i$ is a subset of $A_i$ obtained by sampling each vertex in $A_i$ with probability $2^i p_0$. Thus by Lemma 9, we have

$$\mathbb{E}[|D_i|] = \mathbb{E}[|A_i|] \cdot 2^i p_0 \leq \frac{n}{2^{i-1}} \cdot \frac{2^i}{\sqrt{n}} = 2\sqrt{n} \; .$$ ◀

▶ **Lemma 11.** $D_{\frac{1}{2}\log n} = A_{\frac{1}{2}\log n}$.

**Proof.** We have $2^{\frac{1}{2}\log n} \cdot p_0 \geq 1$. Hence, every vertex of $A_{\frac{1}{2}\log n}$ is included in $D_{\frac{1}{2}\log n}$. ◀

▶ **Lemma 12.** *The expected size of the dominating set returned by the algorithm is at most $O(\sqrt{n}\log n)$ times the optimal size.*

**Proof.** The solution returned by the algorithm is $\cup_{i\geq 0} D_i$ and the set $B \setminus U$. We first observe that $\mathbb{E}[| \cup_{i\geq 0} D_i|] \leq 2\sqrt{n}\log n$ by Lemma 10, so the expected size of this component of the solution is clearly at most $O(\sqrt{n}\log n)$ times the optimal size (assuming the graph is non-empty).

Now to compare $|B \setminus U|$ to optimal solution size, we consider any solution $O^* \subseteq A$ that is a minimum dominating set for $B \setminus U$. We claim that every vertex $a \in O^*$ covers at most $\sqrt{n}\log n$ vertices in $B \setminus U$. Indeed, suppose that this was not the case, and some vertex $a \in O^*$ covers more than $\sqrt{n}\log n$ vertices in $B \setminus U$. Then, there must be a moment when $a$ was inserted into $A_{\frac{1}{2}\log n}$, and thus also into $D_{\frac{1}{2}\log n}$ by Lemma 11. However, from this moment onwards, every uncovered neighbor of $a$ would become covered, a contradiction. Thus $|B \setminus U|$ is bounded by $|O^*|/(\sqrt{n}\log n)$, completing the proof of the lemma. ◀

**High probability result.** The analysis above shows that the expected size of the dominating set produced is at most $O(\sqrt{n}\log n)$ times the optimal size. At the expense of losing another logarithmic factor, it is easy to slightly modify the algorithm and analysis to instead claim that the size of the dominating set produced is at most $O(\sqrt{n}\log^2 n)$ times the optimal size with probability at least $1 - 1/\operatorname{poly}(n)$. The only change in the algorithm is to set the parameter $p_0 = \frac{\log n}{\sqrt{n}}$ instead of $\frac{1}{\sqrt{n}}$. With this change, the assertion of the Lemma 10 can be modified to show that for every integer $i \geq 0$, we have $|D_i| = O(\sqrt{n}\log n)$ with probability at least $1 - 1/\operatorname{poly}(n)$. The remainder of the analysis is essentially identical, and we can conclude that the size of the dominating set produced is at most $O(\sqrt{n}\log^2 n)$ times the optimal size with probability at least $1 - 1/\operatorname{poly}(n)$.

We have thus established the following theorem.

▶ **Theorem 2** (restated and more formal)**.** *Algorithm 1 is a randomized one-pass semi-streaming algorithm for MDS with expected approximation factor $O(\sqrt{n}\log n)$. The algorithm can also give an approximation factor of $O(\sqrt{n}\log^2 n)$ with high probability.*

**Extension to general Set Cover instances.** Our algorithm can also solve Set Cover instances with universe size $n$ and an arbitrary number of sets $m$ with an expected approximation ratio of $O(\sqrt{n}\log m)$ and space $\tilde{O}(n+m)$. The sole modification needed is to initialize $p_0 := \frac{\sqrt{n}}{m}$ in Line 5. Then, similar to Lemma 11, it can be seen that the algorithm only uses at most $\log m$ levels and thus at most $\log m$ dominating sets $D_i$, each of expected size $O(\sqrt{n})$. The expected approximation factor is therefore bounded by $O(\sqrt{n}\log m)$, giving us Theorem 3 (as stated in Section 1.1).

# 4 Lower Bounds

As outlined in the overview presented in Section 1.2, the starting point for our lower bounds for both insertion-only and insertion-deletion streams is the two-party communication problem called the Set-Union problem, whose hardness was established by Assadi et al. [4]. Our lower bound for insertion-deletion streams is based on embedding an instance of Set-Union into a larger instance such that the embedded instance only gets revealed after deletions, allowing us to obtain a stronger lower bound. We refer to this new problem as the Embedded-Set-Union problem (ESU). The rest of the section is organized as follows. In Subsection 4.1, we introduce the Set-Union problem and state its hardness. In Subsection 4.2, we give our lower bound for insertion-only streaming algorithms via a reduction to Set-Union. Then, in Subsection 4.3, we define the Embedded-Set-Union problem, prove its hardness, and establish a connection between insertion-deletion streaming algorithms for MDS and the ESU problem, which yields the desired space lower bound.

## 4.1 The Set-Union Problem

The Set-Union problem is a two-party communication problem where Alice and Bob each hold subsets $S_1, S_2 \subseteq [n]$, respectively, with the promise that $S_1 \cup S_2 = [n] \setminus \{T\}$, for some element $T \in [n]$. The objective for Alice and Bob is to sufficiently *skew* the a priori uniform distribution of $T$ by communicating with each other, i.e., so that the distribution of $T$ conditioned on the transcript of the protocol is far from uniform (see further below for details). While this objective appears to be easier to achieve than identifying $T$ itself, we will see that the information complexity of Set-Union is $\Omega(n)$.

We consider the hard input distribution $\mathcal{D}_{\text{SU}}$ for Set-Union:

Distribution $\mathcal{D}^n_{\text{SU}}$ on variables $A, B$ and $T$:
**1.** For each $i \in [n]$: $(A[i], B[i])$ is chosen uniformly at random from $\{(1,1),(1,0),(0,1)\}$.
**2.** Let $T \in [n]$ be a uniform random index.
**3.** Let $A[T] = B[T] = 0$.
Alice holds $A$ and Bob holds $B$.

If we omit the superscript, i.e., we write $\mathcal{D}_{\text{SU}}$, then we mean $\mathcal{D}^n_{\text{SU}}$.

Following [4] for the related Set-Intersection problem, we consider the following notion of solving Set-Union:

▶ **Definition 13** ([4]). *We say that a protocol $\Pi_{SU}$ $\epsilon$-solves Set-Union iff:*

$$\mathbb{E}_{\Pi_{SU}} \Delta_{TV}\left(dist(T \mid \Pi_{SU}), \mathcal{U}_{[n]}\right) \geq \epsilon \ ,$$

*where $\Delta_{TV}$ is the total variation distance, $T$ is the variable in the distribution $\mathcal{D}_{SU}$, and $\mathcal{U}_{[n]}$ is the uniform distribution on $n$ elements.*

Observe first that $T$ is uniformly distributed in distribution $\mathcal{D}_{\text{SU}}$. The quantity $dist(T \mid \Pi_{\text{SU}})$ is the distribution of $T$ conditioned on the transcript of the protocol. It is hence required that a protocol that $\epsilon$-solves Set-Union contains enough information in the transcript that substantially skews the distribution of $T$.

We will next discuss the hardness of Set-Union. To this end, we observe that Set-Union is the complementary problem to Set-Intersection, as defined by Assadi et al. in [4]: In Set-Intersection, Alice and Bob each hold subsets $R_1, R_2 \subseteq [n]$, respectively, with the promise that $R_1 \cap R_2 = \{T\}$, for some $T \in [n]$. They further define the hard input distribution $\mathcal{D}_{\text{SI}}$, which is identical to $\mathcal{D}_{\text{SU}}$ with every bit flipped. The objective is the same as in Set-Union, i.e., to $\epsilon$-solve Set-Intersection. Since Alice and Bob can take complements of their sets locally, any Set-Union instance $(S_1, S_2)$ can be transformed into a Set-Intersection instance $(R_1, R_2)$ by setting $R_1 = [n] \setminus S_1$ and $R_2 = [n] \setminus S_2$, and vice versa. This process also applies to transforming $\mathcal{D}_{\text{SU}}$ to $\mathcal{D}_{\text{SI}}$. The two problems are therefore equivalent and the hardness of Set-Intersection proved in [4] carries over to Set-Union.

As proved in [4], we have the following hardness:

▶ **Theorem 14** ([4]). *Let $\Pi_{SU}$ be a protocol that $\epsilon$-solves Set-Union. Then:*

$$\text{ICost}^{int}_{\mathcal{D}^n_{SU}}(\Pi_{SU}) = \Omega(\epsilon^2 \cdot n) \ .$$

*By the relationship between internal and external information cost (Lemma 7), the following holds:*

$$\text{ICost}^{ext}_{\mathcal{D}^n_{SU}}(\Pi_{SU}) = \Omega(\epsilon^2 \cdot n) \ .$$

## 4.2 Lower Bound for Insertion-only Streams

We will first use the Set-Union problem to show that every one-pass insertion-only streaming algorithm for MDS with approximation ratio $o(n/\log n)$ can be used to $\Omega(1)$-solve Set-Union. Consequently, any such algorithm requires space $\Omega(n)$. This is achieved via a simple reduction:

▶ **Theorem 4** (restated). *Any constant error one-pass streaming algorithm for MDS in the insertion-only model with approximation factor $o(n/\log n)$ requires $\Omega(n)$ space.*

**Proof.** Let $(A, B, T)$ be an instance of Set-Union, and let $\mathbf{A}$ be a streaming algorithm as in the statement of the theorem.

Alice and Bob proceed as follows:

1. Alice and Bob construct the input graph $G = ([n] \cup \{v_A, v_B\}, E_A \cup E_B)$ as follows:
2. Alice constructs the edge set $E_A = \{(v_A, a) \ : \ a \in A\}$ and runs $\mathbf{A}$ on $E_A$ (in arbitrary order). Alice then sends the memory state of $\mathbf{A}$ to Bob.
3. Bob constructs the edge set $E_B = \{(v_B, b) \ : \ b \in B\}$ and continues executing $\mathbf{A}$ on $E_B$ (in arbitrary order).
4. As a result of $\mathbf{A}$, Bob obtains a dominating set $D$. Bob then sends the set $D' = D \setminus \{v_A, v_B\}$ back to Alice.

We denote this protocol by $\Pi_{\text{SU}}$.

We will now prove that the transcript $\Omega(1)$-solves the Set-Union instance:

First, observe that $\gamma(G) \leq 3$ since the dominating set $\{v_A, v_B, T\}$ is of this size. Next, since the approximation factor of $\mathbf{A}$ is $o(n/\log n)$, the output dominating set $D$ is therefore of size $3 \cdot o(n/\log n) = o(n/\log n)$. Since $T \notin A \cup B$, it is neither covered by $v_A$ nor by $v_B$. $T$

is therefore necessarily contained in the set $D'$, which is part of the transcript. Since $D' \subseteq D$, we have that $|D'| = o(n/\log n)$. Then, since algorithm $\mathbf{A}$ fails with constant error, we have:

$$\mathbb{E}_{\Pi_{\text{SU}}} \Delta_{TV} \left( dist(T \mid \Pi_{\text{SU}}), \mathcal{U}_{[n]} \right) \geq \Pr[\mathbf{A} \text{ succeeds }] \cdot \frac{1}{2} \left( (n - o(n/\log n)) \frac{1}{n} \right) = \Omega(1) \ .$$

Alice and Bob can therefore $\Omega(1)$-solve the Set-Union instance, which, by Theorem 14, implies that the information cost and therefore also the communication cost of $\Pi_{\text{SU}}$ is $\Omega(n)$. Since the message from Bob to Alice, which constitutes the set $D'$, is of length $o(n)$ ($D'$ is of size $o(n/\log n)$ and each element in $D'$ can be encoded with $O(\log n)$ bits), the message from Alice to Bob, which coincides with the memory state of $\mathbf{A}$, must therefore be of length $\Omega(n)$. ◀

## 4.3 Lower Bound for Insertion-deletion Streams

We will now prove our lower bound for insertion-deletion streaming algorithms for MDS, showing that $\Omega(n^2/(\alpha \log n))$ space is necessary for computing an $\alpha$-approximate solution, for any $\alpha = O(n/\log^3 n)$. To this end, we first define a hard two-party communication game Embedded-Set-Union (ESU) in Subsection 4.3.1, which can be seen as the Set-Union problem embedded in a more complex setting, and prove a lower bound on its communication complexity in Subsection 4.3.2. Then, in Subsection 4.3.3, we will show that a streaming algorithm for MDS can be used to solve ESU, which then establishes the desired lower bound.

### 4.3.1 The Embedded-Set-Union Problem (ESU)

The Embedded-Set-Union problem (ESU) is parametrized by an integer $\beta \geq C \log^2 n$, for some large enough constant $C$, and defined as follows. Alice holds $n$ vectors $A_1, \dots, A_n \in \{0,1\}^n$, Bob holds an index $I \in [n]$, a vector $B \in \{0,1\}^n$, index sets $J_1, \dots J_n \subseteq [n]$ with $|J_i| = n - \beta$, for every $i \neq I$ and $J_I = \{\}$, and also knows the entries $A_i[J_i]$, for every $i$. Alice and Bob are guaranteed that sets $(A_I, B)$ constitute a valid Set-Union instance, i.e., $(A_I, B) \sim \mathcal{D}_{\text{SU}}$. The objective for Alice and Bob is to $\epsilon$-solve the Set-Union instance $(A_I, B)$, and we say that a protocol $\epsilon$-solves ESU if it $\epsilon$-solves the instance $(A_I, B)$.

Next, we define a hard input distribution $\mathcal{D}^{\beta}_{\text{ESU}}$ that will be used for lower bounding the communication complexity of ESU.

Distribution $\mathcal{D}^{\beta}_{\text{ESU}}$ is defined on variables $I, T, A_1, \dots A_n$, $J_1, \dots, J_n$, and $B$:
1. Let $I \in [n]$ be a uniform random index.
2. Let $(A_I, B, T) \sim \mathcal{D}_{\text{SU}}(A, B, T)$.
3. For each $i \neq I$ : Let $A_i \sim \mathcal{D}_{\text{SU}}(A)$.
4. Let $J_I = \{\}$ and for $i \neq I$: Let $J_i \subseteq [n]$ be a random subset of size $n - \beta$.

Alice holds sets $A_1, \dots, A_n$ and Bob knows $A_i[J_i]$, for every $i$, the set $B$, and the index $I$.

We will denote by $A'_i := A_i - A_i[J_i]$ the set obtained by removing the elements in $A_i[J_i]$ from $A_i$. In the following, we will require a lower bound on the size of every set $A'_i$ that we will give now:

▶ **Lemma 15.** *Consider an input sampled from $\mathcal{D}^{\beta}_{ESU}$. Then, with high probability, every set $A'_i$ with $i \neq I$, is of size at most $\beta$.*

**Proof.** Since $J_i$ is a uniform random subset of $[n]$ of size $n - \beta$, $A'_i$ is a vector with $\beta$ entries where every bit is "1" with probability $2/3$, since every entry in $A_i$ is set to be 1 with probability $2/3$. Hence, $A'_i$ contains $2/3 \cdot \beta$ "1" entries in expectation, and at most $\beta$ "1" entries with high probability (which follows from Chernoff bounds using the fact that $\beta \geq C \cdot \log n$, for a large enough $C$). Applying the union bound over all $i \in [n] \setminus I$, this bound holds for all indices $i \in [n] \setminus \{I\}$. ◀

In the following, we will work with particular protocols for Set-Union and ESU that we call *nice* protocols:

▶ **Definition 16** (($p, t$)-Nice Protocols). *Let $C_0 > 0$ be a small constant and $n_0$ a large integer such that $\mathrm{ICost}^{ext}_{\mathcal{D}^n_{SU}}(\Pi_{SU}) \geq C_0 \cdot n$, for every $n \geq n_0$, for every protocol $\Pi_{SU}$ that $\frac{1}{4}$-solves Set-Union. We say that a communication protocol $\Pi$ for Set-Union or ESU is $(p,t)$-*nice *if:*

1. *It consists of two messages; first a message $\Pi^1$ from Alice to Bob, and next a message $\Pi^2$ from Bob to Alice; and*
2. *For every $n \geq n_0$, with probability at least $p$, the message $\Pi^2$ is an encoding of a subset $S \subseteq [n]$ with $|S| \leq C_0 \cdot n/(t \log n)$ and $T \in S$. Furthermore, the encoding of $S$, i.e., the message $\Pi^2$, is of length at most $|\Pi^2| \leq C_0 \cdot n/((t-2)\log n)$ bits (which is easy to achieve since every element $u \in S$ can be encoded with $\lceil \log n \rceil$ bits).*

Observe that a $(p,t)$-nice protocol is also $(p',t')$-nice if $p \geq p'$ and $t \geq t'$. For brevity, we say that a protocol is nice if it is $(2/3, 3)$-nice.

We will show now that nice-protocols $\frac{1}{4}$-solve Set-Union (or ESU).

▶ **Lemma 17.** *Let $\Pi$ be a nice protocol for Set-Union (or ESU). Then, for large enough $n$, $\Pi$ $\frac{1}{4}$-solves Set-Union (or ESU).*

**Proof.** Let $C_0$ and $n_0$ be as in Definition 16. Since $\Pi$ is a nice protocol, with probability at least $2/3$, its second message $\Pi^2$ from Bob to Alice encodes a set $S \subseteq [n]$ with $|S| \leq C_0 \cdot n/(3 \log n)$ such that $T \in S$. The transcript of $\Pi$, in particular, the set $S$ thus reduces the possible values of $T$ to $|S| = o(n)$. We thus obtain:

$$\mathbb{E}_{\Pi}\, \Delta_{TV}\left(dist(T \mid \Pi), \mathcal{U}_{[n]}\right) \geq \frac{2}{3} \cdot \frac{1}{2}\left((n - |S|) \cdot \frac{1}{n}\right) = \frac{1}{3}(1 - o(1)) = \frac{1}{3} - o(1) \ . \qquad ◀$$

### 4.3.2 Communication Complexity of ESU

Before giving our lower bound proof, we need to establish an important property of nice protocols for Set-Union.

▶ **Lemma 18.** *Let $\Pi_{SU}$ be a nice protocol for SU, and let $\Pi^1_{SU}$ denote the message sent from Alice to Bob. Then:*

$$I_{\mathcal{D}_{SU}}(A \ : \ \Pi^1_{SU} \mid R_{\Pi_{SU}}) = \Omega(n) \ .$$

**Proof.** Let $C_0$ and $n_0$ be as in Definition 16 and assume that $n \geq n_0$. Then, by Lemma 17, since $\Pi_{\mathrm{SU}}$ is a nice protocol, it $\frac{1}{4}$-solves Set-Union. Recalling the definition of the constant $C_0$, we have $\mathrm{ICost}^{\mathrm{ext}}_{\mathcal{D}^n_{\mathrm{SU}}}(\Pi_{\mathrm{SU}}) \geq C_0 \cdot n$.

We denote by $\Pi^2_{\mathrm{SU}}$ the message sent from Bob to Alice, and, for brevity of notation, we denote by $R$ the public random string $R_{\Pi_{\mathrm{SU}}}$. Then:

$$\begin{aligned}
C_0 \cdot n &\le \mathrm{ICost}^{\mathrm{ext}}_{\mathcal{D}_{\mathrm{SU}}}(\Pi_{\mathrm{SU}}) = I_{\mathcal{D}_{\mathrm{SU}}}(AB \ : \ \Pi_{\mathrm{SU}} \mid R) \\
&= I_{\mathcal{D}_{\mathrm{SU}}}(AB \ : \ \Pi^1_{\mathrm{SU}}\Pi^2_{\mathrm{SU}} \mid R) \\
&= I_{\mathcal{D}_{\mathrm{SU}}}(AB \ : \ \Pi^1_{\mathrm{SU}} | \ R) + I_{\mathcal{D}_{\mathrm{SU}}}(AB \ : \ \Pi^2_{\mathrm{SU}} \mid \Pi^1_{\mathrm{SU}} R) \\
&\le I_{\mathcal{D}_{\mathrm{SU}}}(A \ : \ \Pi^1_{\mathrm{SU}} | \ R) + I_{\mathcal{D}_{\mathrm{SU}}}(B \ : \ \Pi^1_{\mathrm{SU}} \mid AR) + |\Pi^2_{\mathrm{SU}}| \qquad (*) \\
&= I_{\mathcal{D}_{\mathrm{SU}}}(A \ : \ \Pi^1_{\mathrm{SU}} | \ R) + 0 + |\Pi^2_{\mathrm{SU}}| \qquad (**) \\
&\le I_{\mathcal{D}_{\mathrm{SU}}}(A \ : \ \Pi^1_{\mathrm{SU}} | \ R) + \frac{1}{2} C_0 \cdot n \ ,
\end{aligned}$$

where (*) follows since mutual information between two random variables is bounded from above by the minimum entropies of the two involved random variables (Prop. 4 in Sec. 2.2), and (**) follows since $B$ and $\Pi^1_{\mathrm{SU}}$ are independent conditioned on $A$ and $R$ (Prop. 1 in Sec. 2.2). The result follows. ◀

Next, we show that a similar statement also holds for the subvectors of Alice's input $A$.

▶ **Lemma 19.** *Let $\Pi_{SU}$ be a $(2/3+\epsilon, 300)$-nice protocol for* Set-Union*, for any $\epsilon > 0$. Let $J \subseteq [n]$ be a uniform random subset of indices of size $k$, for any $k \le n - C' \log^2 n$ (for some large enough $C'$), and denote by $\overline{J} := [n] \setminus J$. Furthermore, denote by $\Pi^1_{SU}$ the message sent from Alice to Bob. Then:*

$$I_{\mathcal{D}^n_{SU}}(A[\overline{J}] \ : \ \Pi^1_{SU} \mid A[J] R_{\Pi_{SU}}) = \Omega(|\overline{J}|) \ .$$

**Proof.** Let $(A', B', T') \sim \mathcal{D}^{|\overline{J}|}_{\mathrm{SU}}$ be an instance of length $|\overline{J}|$. Alice and Bob use the $(2/3+\epsilon, 300)$-nice protocol $\Pi_{\mathrm{SU}}$ designed for input lengths $n$ to solve this instance of length $|\overline{J}|$. They create the instance $(A, B)$ of length $n$ as follows:

1. Alice and Bob set $A'$ and $B'$ at positions $A[\overline{J}]$ and $B[\overline{J}]$, respectively.
2. Alice and Bob use public randomness to sample all positions $A[j]$ with $j \in J$: For every index $j \in J$, they set $A[j] = 1$ with probability $2/3$ and $A[j] = 0$ with probability $1/3$.
3. Bob uses private randomness to set the bits $B[j]$, for $j \in J$, as follows: If $A[j] = 0$ then Bob sets $B[j] = 1$, and if $A[j] = 1$ then Bob sets $B[j] = 1$ with probability $1/2$ and $B[j] = 0$ with probability $1/2$. Observe that, for every $j \in J$, the pairs $(A[j], B[j])$ are then uniformly distributed in $\{(1,1), (1,0), (0,1)\}$.

Then, Alice sends the first message $\Pi^1_{\mathrm{SU}}$ of protocol $\Pi_{\mathrm{SU}}$ to Bob (on instance $(A, B)$). Bob then computes the second message $\Pi^2_{\mathrm{SU}}$ of protocol $\Pi_{\mathrm{SU}}$ and determines the set $S \subseteq [n]$ encoded by $\Pi^2_{\mathrm{SU}}$. However, rather than sending $\Pi^2_{\mathrm{SU}}$ back to Alice, Bob computes the subset $S' = S \cap \overline{J}$ and sends this subset $S'$ back to Alice in a suitable encoding. We call the resulting protocol $\Pi^{|\overline{J}|}_{\mathrm{SU}}$.

We will first argue that the protocol $\Pi^{|\overline{J}|}_{\mathrm{SU}}$ is $(2/3, 3)$-nice (or simply nice). To this end, we will bound the size of $S'$, i.e., the set of elements sent from Bob back to Alice in $\Pi^{|\overline{J}|}_{\mathrm{SU}}$. Denote by $S$ the set encoded in $\Pi^2_{\mathrm{SU}}$. Then, since $\Pi_{\mathrm{SU}}$ is $(2/3+\epsilon, 300)$-nice, with probability at least $2/3+\epsilon$, $|S| \le C_0 \cdot n/(300 \log n)$ and $T \in S$. Recall that $S' = S \cap \overline{J}$. Then, since $\overline{J}$ is a random subset of $[n]$, by Chernoff bounds, we have

$$|S'| \le 100 \cdot \frac{C_0 \cdot n}{300 \log n} \cdot \frac{|\overline{J}|}{n} = \frac{C_0 \cdot |\overline{J}|}{3 \log n} \ ,$$

with probability at least $1 - \frac{1}{n}$ (using the fact that $|\overline{J}| \geq C' \log^2 n$, for some large enough $C'$), and $T \in S'$. Using a union bound on the error probabilities $1/3 - \epsilon$ and $1/\operatorname{poly} n$, the protocol $\Pi_{\text{SU}}^{|\overline{J}|}$ is thus $(2/3, 3)$-nice (for large enough $n$).

Next, we will relate the information cost of $\Pi_{\text{SU}}$ to $\Pi_{\text{SU}}^{\overline{J}}$. Observe that the public randomness $R_{\Pi_{\text{SU}}^{|\overline{J}|}}$ used in $\Pi_{\text{SU}}^{|\overline{J}|}$ is $R_{\Pi_{\text{SU}}^{|\overline{J}|}} = A[J], J, R_{\Pi_{\text{SU}}}$, where $R_{\Pi_{\text{SU}}}$ is the public randomness used by protocol $\Pi_{\text{SU}}$. Observe further that, since $\overline{J}$ is a random subset, this construction establishes exactly the distribution $\mathcal{D}_{\text{SU}}^n$. Denote by $(\Pi_{\text{SU}}^{|\overline{J}|})^1$ the message sent from Alice to Bob in $\Pi_{\text{SU}}^{|\overline{J}|}$. Then, since $\Pi_{\text{SU}}^{|\overline{J}|}$ is $(2/3, 3)$-nice, we can apply Lemma 18 and obtain:

$$\begin{aligned}\Omega(|\overline{J}|) &= I_{\mathcal{D}_{\text{SU}}^{|\overline{J}|}}(A' \ : \ (\Pi_{\text{SU}}^{|\overline{J}|})^1 \mid R_{\Pi_{\text{SU}}^{|\overline{J}|}}) \\ &= I_{\mathcal{D}_{\text{SU}}^n}(A[\overline{J}] \ : \ \Pi_{\text{SU}}^1 \mid A[J] J R_{\Pi_{\text{SU}}}) \\ &= I_{\mathcal{D}_{\text{SU}}^n}(A[\overline{J}] \ : \ \Pi_{\text{SU}}^1 \mid A[J] R_{\Pi_{\text{SU}}}) \ ,\end{aligned}$$

where the last step uses the fact that $J$ is independent of all other variables in the mutual information term (Prop. 3 in Sec. 2.2). ◀

Equipped with Lemma 19, we are now ready to give our lower bound on the communication complexity of ESU.

▶ **Theorem 20.** *Every $(2/3 + \epsilon, 300)$-nice protocol for ESU requires a first message from Alice to Bob of size $\Omega(n\beta)$ bits.*

**Proof.** Let $\Pi_{\text{ESU}}$ be a $(2/3 + \epsilon, 300)$-nice protocol for ESU. We will first show that $\Pi_{\text{ESU}}$ can be used to solve Set-Union, as follows:

1. Let $(A, B, T) \sim \mathcal{D}_{\text{SU}}$ be an instance of Set-Union.
2. Alice and Bob use public randomness to sample a uniform random index $I \in [n]$.
3. Alice sets $A_I = A$.
4. For every $i \neq I$:
   Using public randomness, Alice and Bob sample $J_i \subseteq [n]$ of size $n - \beta$ and $A_i[J_i]$ (each bit in $A_i[J_i]$ is 1 with probability 2/3, otherwise 0). Then, using private randomness, Alice samples the remaining positions of $A_i$, namely, $A_i[\overline{J_i}]$.

Next, they run the protocol $\Pi_{\text{ESU}}$, which solves ESU and thus also Set-Union. We denote the resulting protocol by $\Pi_{\text{SU}}$. Observe that since $\Pi_{\text{ESU}}$ is $(2/3 + \epsilon, 300)$-nice, the protocol $\Pi_{\text{SU}}$ is also $(2/3 + \epsilon, 300)$-nice.

Let $\overline{J_I} \subseteq [n]$ be a random subset of size $\beta$ and let $J_I = [n] - \overline{J_I}$. Then, using Lemma 19, we obtain (recall that $\beta \geq C \log^2 n$, for some large enough $C$, we can thus invoke Lemma 19):

$$\begin{aligned}\Omega(\beta) \quad &= \quad I_{\mathcal{D}_{\text{SU}}}(A[\overline{J_I}] \ : \ \Pi_{\text{SU}}^1 \mid A[J_I] R_{\Pi_{\text{SU}}}) \\ &= \quad I_{\mathcal{D}_{\text{ESU}}}(A_I[\overline{J_I}] \ : \ \Pi_{\text{ESU}}^1 \mid I A[J_I] A[J_{-I}] R_{\Pi_{\text{ESU}}}) \\ &= \quad I_{\mathcal{D}_{\text{ESU}}}(A_I[\overline{J_I}] \ : \ \Pi_{\text{ESU}}^1 \mid I A[J] R_{\Pi_{\text{ESU}}}) \ ,\end{aligned}$$

where we used the notation $A[J_{-I}] = A_1[J_1], \ldots, A_{I-1}[J_{I-1}], A_{I+1}[J_{I+1}], \ldots, A_n[J_n]$, and $A[J] = A_1[J_1], \ldots, A_n[J_n]$, and the fact that the public randomness $R_{\Pi_{\text{SU}}}$ consists of $I$, $A[J_{-I}]$, and $R_{\Pi_{\text{ESU}}}$.

Next, we expand the previous expression as follows:

$$
\begin{aligned}
&\Omega(\beta)\\
&= I_{\mathcal{D}_{\mathrm{ESU}}}(A_I[\overline{J_I}] \ : \ \Pi^1_{\mathrm{ESU}} \mid IA[J]R_{\Pi_{\mathrm{ESU}}})\\
&= \frac{1}{n} \cdot \sum_{i=1}^{n} I_{\mathcal{D}_{\mathrm{ESU}}}(A_i[\overline{J_i}] \ : \ \Pi^1_{\mathrm{ESU}} \mid IA[J]R_{\Pi_{\mathrm{ESU}}}, I = i)\\
&= \frac{1}{n} \cdot \sum_{i=1}^{n} I_{\mathcal{D}_{\mathrm{ESU}}}(A_i[\overline{J_i}] \ : \ \Pi^1_{\mathrm{ESU}} \mid A[J]R_{\Pi_{\mathrm{ESU}}}) && \text{Prop. 3 in Sec. 2.2 applied}\\
& && \text{to } I \text{ (and event } I = i)\\
&\le \frac{1}{n} \cdot \sum_{i=1}^{n} I_{\mathcal{D}_{\mathrm{ESU}}}(A_i[\overline{J_i}] \ : \ \Pi^1_{\mathrm{ESU}} \mid A[J]R_{\Pi_{\mathrm{ESU}}}, A_1[\overline{J_1}], \dots, A_{i-1}[\overline{J_{i-1}}]) && \text{Prop. 2 in Sec. 2.2}\\
&= \frac{1}{n} I_{\mathcal{D}_{\mathrm{ESU}}}(A[\overline{J}] \ : \ \Pi^1_{\mathrm{ESU}} \mid A[J]R_{\Pi_{\mathrm{ESU}}})\\
&\le \frac{1}{n} I_{\mathcal{D}_{\mathrm{ESU}}}(A \ : \ \Pi^1_{\mathrm{ESU}} \mid R_{\Pi_{\mathrm{ESU}}}) \ .
\end{aligned}
$$

Since $I_{\mathcal{D}_{\mathrm{ESU}}}(A \ : \ \Pi^1_{\mathrm{ESU}} \mid R_{\Pi_{\mathrm{ESU}}}) \le H_{\mathcal{D}_{\mathrm{ESU}}}(\Pi^1_{\mathrm{ESU}}) \le |\Pi^1_{\mathrm{ESU}}|$, the result follows. ◀

### 4.3.3 Lower Bound for Insertion-deletion Streaming Algorithms for MDS

We will now argue that one-pass insertion-deletion streaming algorithms for MDS can be used to obtain a $(p,t)$-nice protocol for ESU, for suitable parameters $p$ and $t$. This reduction then reveals that every such algorithm requires space $\Omega(n^2/(\alpha \log n))$, which constitutes our main lower bound result.

▶ **Theorem 5** (restated and slightly more formal). *Every one-pass $1/4$-error streaming algorithm in the insertion-deletion model for MDS with approximation factor $\alpha \le Cn/\log^3 n$, for some small enough $C$, requires space $\Omega(n^2/(\alpha \log n))$.*

**Proof.** Let $\mathbf{A}$ be an algorithm as in the statement of the theorem. We consider the ESU problem with parameter $\beta = C' \frac{n}{\alpha \lceil \log n \rceil}$, for a sufficiently small constant $C'$ whose value we determine later. Recall that the definition of ESU requires $\beta \ge C \log^2 n$, for some large enough constant $C$, which, together with $\beta = C' \frac{n}{\alpha \lceil \log n \rceil}$ implies that $\alpha \le C'' \frac{n}{\log^3 n}$, for some constant $C''$. This theorem thus holds for approximation factors $\alpha \le C'' \frac{n}{\log^3 n}$.

Let $(A_1, \dots, A_n, I, B, J_1, \dots, J_n) \sim \mathcal{D}^{\beta}_{\mathrm{ESU}}$ be an input to ESU. Then:

1. Alice and Bob construct a graph $G$ on vertex set $V = \{A_1, \dots, A_n, B\} \cup [n]$, where $A_1, \dots, A_n, B$ is turned into a clique. Denote by $E_C$ the edges of this clique. Alice runs algorithm $\mathbf{A}$ on the edges in $E_C$ (in arbitrary order).
2. Alice constructs the edge set $E_A = \{(A_i, j) \ : \ i \in [n] \text{ and } j \in A_i\}$. Alice runs algorithm $\mathbf{A}$ on these edges (in arbitrary order) and then sends the resulting memory state to Bob.
3. Bob constructs the edge set $E_B = \{(B, j) \ : \ j \in B\}$ and continues the execution of $\mathbf{A}$ on the edges in $E_B$ (in arbitrary order).
4. Bob next constructs the following set of edge deletions: $E_D = \{(A_i, j) \ : \ i \in [n] \text{ and } j \in A_i[J_i]\}$. Bob continues the execution of $\mathbf{A}$ on $E_D$ (in arbitrary order) with every edge interpreted as an edge deletion.
5. Bob then examines the dominating set $D$ and the cover certificate $C$ output by $\mathbf{A}$. Denote by $F$ all elements in $[n]$ that are not covered by the sets $A_I, B$. Bob sends $F$ back to Alice.

We denote this protocol by $\Pi_{\text{ESU}}$ and claim that $\Pi_{\text{ESU}}$ is $(3/4, 300)$-nice. Indeed, first observe that $\gamma(G) \leq 3$ since $\{A'_I, B, \{T\}\}$ is a dominating set of this size. Next, observe that $T \in F$, i.e., $T$ is necessarily covered by a set other than the sets $A'_I$ and $B$ since $(A'_I, B, T)$ constitutes a Set-Union instance. By Lemma 15, the degree of every vertex in $V \setminus \{A'_I, B\}$ is at most $\beta$ w.h.p. Hence, these vertices cover at most $|D| \cdot \beta$ different elements, which implies $|F| \leq |D| \cdot \beta$. Since $D$ constitutes an $\alpha$-approximation, we have $|D| \leq 3 \cdot \alpha$. Thus there are at most $|F| \leq 3 \cdot \alpha \cdot \beta \leq \frac{3C'n}{\lceil \log n \rceil}$ values that variable $T$ could take on. We then select $C'$ such that $\frac{3C'n}{\lceil \log n \rceil} \leq \frac{C_0 n}{3 \log n}$, where $C_0$ is as in Definition 16. Finally, since the success probability of A is at least 3/4, and $|F| \leq |D| \cdot \beta$ holds with high probability, we obtain that $\Pi_{\text{ESU}}$ is nice.

Now by Theorem 20, the first message sent in protocol $\Pi_{\text{ESU}}$ must be of size $\Omega(n\beta) = \Omega(n^2/(\alpha \log n))$. Since this message coincides with the space requirements of **A**, the result follows. ◀

## 5 Conclusions

Our results resolve the space requirements of streaming algorithms for MDS in both the insertion-only and the insertion-deletion models up to poly-logarithmic factors. We showed that, similar to Set Cover, MDS undergoes a phase transition at an approximation factor of $\tilde{\Theta}(\sqrt{n})$, where space $\tilde{\mathrm{O}}(n)$ is sufficient for computing an $\tilde{\mathrm{O}}(\sqrt{n})$-approximation, but space $\Omega(n\sqrt{n})$ is needed for obtaining a $o(\sqrt{n})$-approximation. We also showed that no such transition occurs in the insertion-deletion model, where space $\tilde{\Theta}(n^2/\alpha)$ is necessary and sufficient for computing an $\alpha$-approximation. We conclude with a discussion of two natural questions suggested by our work.

First, while our lower bound in the insertion-only model does not require an algorithm to output a cover certificate, the cover certificate is pivotal to the lower bound by Assadi et al. for Set-Cover [5] as well as our lower bound for MDS in the insertion-deletion model. Can we prove similar lower bounds for algorithms that are not required to output a cover certificate?

Second, as observed in Section 3, our insertion-only streaming algorithm naturally extends to arbitrary Set-Cover instances in the edge-arrival model, and gives $\tilde{\mathrm{O}}(\sqrt{n})$-approximation with space $\tilde{\mathrm{O}}(m+n)$ where $m$ denotes the number of sets and $n$ denotes the number of elements in the set cover instance. In the set-arrival model, i.e., when entire sets arrive one-by-one, a similar approximation factor can be achieved using only $\tilde{\mathrm{O}}(n)$ space, without any dependency on $m$ [14, 9]. This suggests the following natural question: is it possible to avoid the space dependency on $m$ in the edge-arrival model? In a follow-up work, subsequent to this submission, we have made progress on this question by showing that the space used must necessarily have a dependence on $m$. Specifically, we have been able to show that $\tilde{\Omega}(m+n)$ space is necessary for achieving an $\tilde{\mathrm{O}}(\sqrt{n})$-approximation.

## References

1 Kook Jin Ahn, Sudipto Guha, and Andrew McGregor. Analyzing graph structure via linear measurements. In Yuval Rabani, editor, *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 459–467. SIAM, 2012. doi:10.1137/1.9781611973099.40.

2 Kook Jin Ahn, Sudipto Guha, and Andrew McGregor. Spectral sparsification in dynamic graph streams. In Prasad Raghavendra, Sofya Raskhodnikova, Klaus Jansen, and José D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques - 16th International Workshop, APPROX 2013, and 17th International Workshop,*

*RANDOM 2013, Berkeley, CA, USA, August 21-23, 2013. Proceedings*, volume 8096 of *Lecture Notes in Computer Science*, pages 1–10. Springer, 2013. doi:10.1007/978-3-642-40328-6_1.

3 Sepehr Assadi. Tight space-approximation tradeoff for the multi-pass streaming set cover problem. In Emanuel Sallinger, Jan Van den Bussche, and Floris Geerts, editors, *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2017, Chicago, IL, USA, May 14-19, 2017*, pages 321–335. ACM, 2017. doi:10.1145/3034786.3056116.

4 Sepehr Assadi, Yu Chen, and Sanjeev Khanna. Polynomial pass lower bounds for graph streaming algorithms. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2019, pages 265–276, New York, NY, USA, 2019. Association for Computing Machinery. doi:10.1145/3313276.3316361.

5 Sepehr Assadi, Sanjeev Khanna, and Yang Li. Tight bounds for single-pass streaming complexity of the set cover problem. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 698–711. ACM, 2016. doi:10.1145/2897518.2897576.

6 Sepehr Assadi, Sanjeev Khanna, Yang Li, and Grigory Yaroslavtsev. Maximum matchings in dynamic graph streams and the simultaneous communication model. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1345–1364. SIAM, 2016. doi:10.1137/1.9781611974331.ch93.

7 Michael Barlow, Christian Konrad, and Charana Nandasena. Streaming set cover in practice. In Martin Farach-Colton and Sabine Storandt, editors, *Proceedings of the Symposium on Algorithm Engineering and Experiments, ALENEX 2021, Virtual Conference, January 10-11, 2021*, pages 181–192. SIAM, 2021. doi:10.1137/1.9781611976472.14.

8 Mark Braverman. Interactive information complexity. *SIAM Rev.*, 59(4):803–846, 2017. doi:10.1137/17M1139254.

9 Amit Chakrabarti and Anthony Wirth. Incidence geometries and the pass complexity of semi-streaming set cover. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1365–1373. SIAM, 2016. doi:10.1137/1.9781611974331.ch94.

10 Graham Cormode, Howard J. Karloff, and Anthony Wirth. Set cover algorithms for very large datasets. In Jimmy Huang, Nick Koudas, Gareth J. F. Jones, Xindong Wu, Kevyn Collins-Thompson, and Aijun An, editors, *Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, Toronto, Ontario, Canada, October 26-30, 2010*, pages 479–488. ACM, 2010. doi:10.1145/1871437.1871501.

11 Thomas M. Cover and Joy A. Thomas. *Elements of information theory (2. ed.)*. Wiley, 2006. URL: http://www.elementsofinformationtheory.com/.

12 Jacques Dark and Christian Konrad. Optimal lower bounds for matching and vertex cover in dynamic graph streams. In Shubhangi Saraf, editor, *35th Computational Complexity Conference, CCC 2020, July 28-31, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 169 of *LIPIcs*, pages 30:1–30:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.CCC.2020.30.

13 Erik D. Demaine, Piotr Indyk, Sepideh Mahabadi, and Ali Vakilian. On streaming and communication complexity of the set cover problem. In Fabian Kuhn, editor, *Distributed Computing - 28th International Symposium, DISC 2014, Austin, TX, USA, October 12-15, 2014. Proceedings*, volume 8784 of *Lecture Notes in Computer Science*, pages 484–498. Springer, 2014. doi:10.1007/978-3-662-45174-8_33.

14 Yuval Emek and Adi Rosén. Semi-streaming set cover. *ACM Trans. Algorithms*, 13(1):6:1–6:22, 2016. doi:10.1145/2957322.

15 Joan Feigenbaum, Sampath Kannan, Andrew McGregor, Siddharth Suri, and Jian Zhang. On graph problems in a semi-streaming model. *Theor. Comput. Sci.*, 348(2-3):207–216, 2005. doi:10.1016/j.tcs.2005.09.013.

16 Sariel Har-Peled, Piotr Indyk, Sepideh Mahabadi, and Ali Vakilian. Towards tight bounds for the streaming set cover problem. In Tova Milo and Wang-Chiew Tan, editors, *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, pages 371–383. ACM, 2016. doi:10.1145/2902251.2902287.

17 Monika Rauch Henzinger, Prabhakar Raghavan, and Sridhar Rajagopalan. Computing on data streams. In James M. Abello and Jeffrey Scott Vitter, editors, *External Memory Algorithms, Proceedings of a DIMACS Workshop, New Brunswick, New Jersey, USA, May 20-22, 1998*, volume 50 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 107–118. DIMACS/AMS, 1998. doi:10.1090/dimacs/050/05.

18 Piotr Indyk, Sepideh Mahabadi, Ronitt Rubinfeld, Jonathan R. Ullman, Ali Vakilian, and Anak Yodpinyanee. Fractional set cover in the streaming model. In Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017, August 16-18, 2017, Berkeley, CA, USA*, volume 81 of *LIPIcs*, pages 12:1–12:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.APPROX-RANDOM.2017.12.

19 Hossein Jowhari, Mert Saglam, and Gábor Tardos. Tight bounds for lp samplers, finding duplicates in streams, and related problems. In Maurizio Lenzerini and Thomas Schwentick, editors, *Proceedings of the 30th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2011, June 12-16, 2011, Athens, Greece*, pages 49–58. ACM, 2011. doi:10.1145/1989284.1989289.

20 Michael Kapralov, Yin Tat Lee, Cameron Musco, Christopher Musco, and Aaron Sidford. Single pass spectral sparsification in dynamic streams. *SIAM J. Comput.*, 46(1):456–477, 2017. doi:10.1137/141002281.

21 Michael Kapralov, Aida Mousavifar, Cameron Musco, Christopher Musco, Navid Nouri, Aaron Sidford, and Jakab Tardos. Fast and space efficient spectral sparsification in dynamic streams. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 1814–1833. SIAM, 2020. doi:10.1137/1.9781611975994.111.

22 Christian Konrad. Maximum matching in turnstile streams. In Nikhil Bansal and Irene Finocchi, editors, *Algorithms - ESA 2015 - 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings*, volume 9294 of *Lecture Notes in Computer Science*, pages 840–852. Springer, 2015. doi:10.1007/978-3-662-48350-3_70.

23 Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, 1997.

24 Anup Rao and Amir Yehudayoff. *Communication Complexity: and Applications*. Cambridge University Press, 2020. doi:10.1017/9781108671644.

25 Tim Roughgarden. Communication complexity (for algorithm designers). *Foundations and Trends® in Theoretical Computer Science*, 11(3–4):217–404, 2016. doi:10.1561/0400000076.

26 Barna Saha and Lise Getoor. 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*, pages 697–708. SIAM, 2009. doi:10.1137/1.9781611972795.60.

## A The AKL-Algorithm

For completeness, we now discuss the AKL-algorithm [5] applied to MDS in data streams. We will first present the algorithm applied to insertion-only streams and then introduce the modifications necessary for it to be run in insertion-deletion streams.

Let $G = (V, E)$ be the input graph to MDS, and let $\alpha \geq 1$ be the desired approximation guarantee. The AKL-algorithm in insertion-only streams proceeds as follows:

1. **Preprocessing:** Arbitrarily partition $V$ into sets $\mathcal{V} = \{V_1, V_2, \ldots, V_{\lceil n/\alpha \rceil}\}$ such that $|V_i| \leq \alpha$, for every $i$.
2. **While processing the input edge stream:** For every vertex $v \in V$ and every vertex group $V_j \in \mathcal{V}$, store a single edge that connects $v$ to an arbitrary vertex in $V_j$. Denote by $S$ the set of edges stored.
3. **Postprocessing:**
   a. Using edge set $S$, construct the split graph $G' = (V \cup \mathcal{V}, E')$ where $\mathcal{V}$ is turned into a clique, and there is an edge between $v \in V$ and $V_j \in \mathcal{V}$ if $S$ contains an edge $(v, u)$, for any $u \in V_j$. Compute a minimum dominating set $D'$ in $G'$ with the property that no vertex in $V$ is selected (in exponential time).
   b. Define the output dominating set $D$ as follows: For each vertex group $V_j \in D'$, add all vertices in $V_j$ to $D$.

**Analysis.** The space requirements of the algorithm are dominated by the edges $S$ retained by the algorithm. Since, for each vertex $v \in V$, we store at most $\lceil n/\alpha \rceil$ incident edges, and we account space $O(\log n)$ per edge, the total space used by the algorithm is $O(n \cdot \frac{n}{\alpha} \cdot \log n) = O(\frac{n^2}{\alpha} \log n)$.

To see that the algorithm indeed has an approximation factor of $\alpha$, we will argue that $|D'| = \gamma(G') \leq \gamma(G)$. This then implies the result since, by construction, $|D| \leq \alpha \cdot |D'|$ and thus $|D| \leq \alpha \cdot \gamma(G)$.

To see that $\gamma(G') \leq \gamma(G)$ holds, consider the split graph $G'' = (V_1 \cup V_2, E'')$, where $V_1$ and $V_2$ are copies of $V$, $V_1$ is turned into a clique and $(v_1, v_2) \in E''$ iff $v_1$ and $v_2$ are neighbors in $G$. Then it is not hard to see that $\gamma(G) = \gamma(G'')$ (a dominating set $Q$ in $G$ is also a dominating set in $G''$ if $Q$ is regarded as a subset of $V_1$, and vice versa). Next, observe that $G'$ can be obtained from $G''$ by contracting the $V_2$ vertices into the vertex groups $\mathcal{V}$. Since the domination number of a graph cannot increase when contracting vertices, we have $\gamma(G') \leq \gamma(G'') = \gamma(G)$.

Last, we need to argue that set $D$ is indeed a dominating set in $G$. Recall that, by construction, $D'$ is a dominating set in $G'$. Consider a vertex $v \in V$ and a vertex group $V_j \in \mathcal{V} \cap D'$ that dominates $v$ in $G'$. This implies that there exists a vertex in $V_j$ that is adjacent to $v$ in $G$. Since we added all vertices of $V_j$ to $D$, vertex $v$ is thus also dominated by a vertex in $D$ in graph $G$.

We have thus established the following theorem:

▶ **Theorem 21.** *For every $1 \leq \alpha \leq n$, there is a deterministic one-pass $\alpha$-approximation streaming algorithm in the insertion-only model with space $O(\frac{n^2}{\alpha} \log n)$.*

**Extension to Insertion-deletion Streams.** In order to implement the AKL-algorithm in the insertion-deletion model, we need to solve the following task: For every $v \in V$ and vertex group $V_j \in \mathcal{V}$, we need to store an arbitrary edge that connects $v$ to a vertex in $V_j$ (if there is one). This can be achieved by $l_0$-*sampling*: An $l_0$-sampler in insertion-deletion streams is an algorithm that returns a uniform random element with non-zero frequency, which, applied to insertion-deletion graph streams, is thus able to return a uniform random edge of the input graph. By restricting the scope of the $l_0$-sampler to edges between vertex $v \in V$ and vertex group $V_j \in \mathcal{V}$, $l_0$-sampling allows us to sample a uniform random edge connecting $v$ to (a vertex in) $V_j$.

The $l_0$-samplers of Jowhari et al. [19] require space $O(\log^2 n \log \frac{1}{\delta})$, where $\delta$ is the success probability of the sampler. Since we need to run an $l_0$-sampler for every pair $(v, V_j) \in V \times \mathcal{V}$, which amounts to $\Theta(n^2/\alpha)$ samplers, we choose $\delta = \Theta(\frac{1}{n^2})$ in order to obtain an algorithm that succeeds with high probability. The total space requirements of the algorithm are dominated by the space required by the $l_0$-samplers, which amount to

$$O(\frac{n^2}{\alpha}) \cdot O(\log^2 n \log \frac{1}{\delta}) = O(\frac{n^2}{\alpha} \log^3 n) \ .$$

We thus established the following theorem:

▶ **Theorem 22.** *For every $1 \leq \alpha \leq n$, there is a randomized one-pass $\alpha$-approximation streaming algorithm in the insertion-deletion model with space $O(\frac{n^2}{\alpha} \log^3 n)$ that succeeds with high probability.*