Deterministic Independent Sets in the Semi-Streaming Model

Daniel Ye
University of Waterloo, Canada

Abstract

We consider the independent set problem in the semi-streaming model. For any input graph $G = (V, E)$ with n vertices, an independent set is a set of vertices with no edges between any two elements. In the semi-streaming model, G is presented as a stream of edges and any algorithm must use $\widetilde{O}(n)$[1] bits of memory to output a large independent set at the end of the stream.

Prior work has designed various semi-streaming algorithms for finding independent sets. Due to the hardness of finding maximum and maximal independent sets in the semi-streaming model, the focus has primarily been on finding independent sets in terms of certain parameters, such as the maximum degree Δ. In particular, there is a simple randomized algorithm that obtains independent sets of size $\frac{n}{\Delta+1}$ in expectation, which can also be achieved with high probability using more complicated algorithms. For deterministic algorithms, the best bounds are significantly weaker. The best we know is a straightforward algorithm that finds an $\tilde{\Omega}\left(\frac{n}{\Delta^2}\right)$ size independent set.

We show that this straightforward algorithm is nearly optimal by proving that any deterministic semi-streaming algorithm can only output an $\widetilde{O}\left(\frac{n}{\Delta^2}\right)$ size independent set. Our result proves a strong separation between the power of deterministic and randomized semi-streaming algorithms for the independent set problem.

2012 ACM Subject Classification Theory of computation → Lower bounds and information complexity

Keywords and phrases Sublinear Algorithms, Derandomization, Semi-Streaming Algorithms

Digital Object Identifier 10.4230/LIPIcs.ICALP.2025.135

Category Track A: Algorithms, Complexity and Games

Funding Supported in part by Sepehr Assadi's Sloan Research Fellowship and NSERC Discovery grant.

Acknowledgements I would like to thank Sepehr Assadi for introducing me to this problem and for his numerous discussions, comments, and feedback throughout the development of this work. Without him, this would not have been possible. I would also like to thank Vihan Shah, Janani Sundaresan, and Christopher Trevisan for their insightful comments and meticulous remarks on earlier versions of this work.

1 Introduction

Finding an independent set of a graph is a classical problem in graph theory with wide-ranging applications. An independent set of $G = (V, E)$ is a subset $I \subseteq V$ such that none of the vertices in I have an edge between them. Being able to find large independent sets plays a crucial role in network scheduling, transportation management, and much more. Especially in recent years, there has been an increased need for handling massive graphs to solve various tasks. Consequently, there has been an increased interest in studying the independent set problem in modern models of computation, such as the semi-streaming model introduced

[1] $\widetilde{O}(\cdot)$ is used to hide polylog factors.

52nd International Colloquium on Automata, Languages, and Programming (ICALP 2025).
Editors: Keren Censor-Hillel, Fabrizio Grandoni, Joël Ouaknine, and Gabriele Puppis
Article No. 135; pp. 135:1–135:17
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

in [14]. In this model, G is presented as a stream of edges and the storage space of the algorithm is bounded to $\widetilde{O}(n)$, where $n = |V|$. Our result pertains to the independent set problem under the semi-streaming model.

It is known that the Maximum Independent Set problem is both NP-hard [18] and hard-to-approximate to within a factor of $n^{1-\delta}$ for any $\delta > 0$ [20]. Not only that, it is hard-to-approximate in the semi-streaming model, regardless of the time complexity of the algorithm [17]. As such, one line of research has focused on obtaining *Maximal* Independent Set. In this respect, [2] yields an $O(\log \log n)$-pass semi-streaming algorithm for this problem, and very recently, [5] proved this is the optimal number of passes.

Hence, to study algorithms in even fewer passes, the problem must be further relaxed to finding "combinatorially optimal" independent sets. On one hand, we can try finding an optimal bound with respect to degree sequences – the Caro-Wei theorem guarantees an independent set of size $\sum_v \frac{1}{1+\deg(v)}$, which is optimal in the sense that there are many graphs that do not admit larger independent sets. To achieve this bound in the semi-streaming model, [15] devises an algorithm for hypergraphs, which when applied to graphs, yields an expected independent set of size $\Omega\left(\sum_v \frac{1}{\deg(v)}\right)$ with $O(n)$ memory and $O(1)$ time per edge.

On the other hand, if we consider bounds with respect to n and Δ (the maximum degree in a graph), the Caro-Wei theorem guarantees independent sets of size $\frac{n}{\Delta+1}$, which is also tight. There is a very easy randomized semi-streaming algorithm for achieving this bound in expectation: randomly permute the vertices, and when an edge arrives in the stream, mark the endpoint that appears later. At the end, output all unmarked vertices. It is a standard result in graph theory that this algorithm outputs an independent set of size $\frac{n}{\Delta+1}$. We can even obtain such an independent set with high probability using the $(\Delta + 1)$-coloring semi-streaming algorithm presented in [4].

Interestingly, these results all make heavy use of randomization. Thus, as with a plethora of other problems in the semi-streaming model, there is a particular interest in derandomizing algorithms to achieve similar performance. Some problems admit single-pass deterministic algorithms matching their randomized counterparts (e.g. connectivity and bipartiteness [1], maximum matching [19]). However, others have a proven discrepancy between the theoretical space complexities of deterministic algorithms and randomized ones under this model (e.g. triangle counting [8] and vertex coloring [3]).

For the independent set problem, the state-of-the-art has been a fairly simple deterministic algorithm for finding an independent set of size $\frac{n}{\Delta^2}$. We begin by choosing $\frac{n}{\Delta}$ vertices arbitrarily, then store all the edges between them (resulting in $O\left(\frac{n}{\Delta} \cdot \Delta\right) = O(n)$ edges). We can then calculate a maximal independent set of this subgraph, resulting in an independent set of size $O\left(\frac{n}{\Delta^2}\right)$. There has been very little progress in constructing a deterministic algorithm that does better than this, which has led to the following open question:

> *Is there a single-pass semi-streaming deterministic algorithm that can match the performance of randomized ones? In particular, is there a deterministic algorithm that can find an independent set of size better than $\widetilde{O}\left(\frac{n}{\Delta^2}\right)$ in general graphs?*

1.1 Our Contributions

Our main result is a negative answer to the open question: the deterministic algorithm stated above is optimal up to polylog factors.

▶ **Result 1.** *Any deterministic single-pass semi-streaming algorithm can only find an independent set of size at most $\widetilde{O}\left(\frac{n}{\Delta^2}\right)$ in a graph of maximum degree Δ (even when Δ is known).*

To the best of our knowledge, no space lower bounds have been devised for deterministic algorithms solving the independent set problem in the semi-streaming model. This result provides a lower bound that is tight up to logarithmic factors, illustrating another significant separation between deterministic and randomized algorithms in the semi-streaming model.

1.2 Our Techniques

We give a summary of our techniques here.

We begin our proof of the space lower bound in Result 1 with a similar setup to [3], who derived a space lower bound for deterministic algorithms solving the vertex coloring problem. In our case, we consider the multi-party communication complexity of the Independent Set problem, where the edges of a graph are partitioned among some number of players. In a predefined order, each player may speak once (outputting $\widetilde{O}(n)$ bits) and is heard by all future players. The goal is for the last player to output a large independent set.

We will design an adversary that adaptively constructs a graph that forces the algorithm to output an independent set of size $\widetilde{O}\left(\frac{n}{\Delta^2}\right)$. To do so, it is useful to consider the graph of non-edges (this is slightly *different* from the missing graph we use in our formal proof). For each player, its non-edges are determined by the messages of all the previous players. We first consider the set of inputs consistent with the previous players' messages, then define the non-edges as the set of all edges that were *not* sent to any previous player in any consistent input. This is useful because any independent set the last player outputs must be a clique in its graph of non-edges (otherwise, there would be some input graph consistent with all the messages that renders the algorithm incorrect).

Additionally, we make use of the compression lemma of [3]. In particular, on any arbitrary graph with degree d, we can create a distribution of graphs by sampling each edge with probability p while removing results with degree $\geq 2pd$. Then, for any algorithm that compresses a graph in this distribution to an s-bit summary, the compression lemma finds a summary such that at most $O\left(\frac{s}{p}\right)$ edges are not in any graph mapped to that summary.

This is useful for [3] as their adversary can narrow its search to a small set of vertices that a deterministic algorithm cannot summarize well. In fact, by focusing on vertices with low non-edge degrees, it can sample each remaining edge with a higher probability and improve the bound given by the compression lemma. However, for the independent set problem, deterministic algorithms are free to choose vertices from *any* section of the graph (and do not need to deal with *every* vertex as in coloring). As such, our adversary does not have the luxury of searching for some useful set of vertices nor working with vertices with low non-edge degrees. Instead, it must remove large independent sets globally from the graph and account for vertices that might not be easy to work with. To achieve this, our adversary generates graphs that are similar in structure to a Turán graph. In particular, as a deterministic algorithm receives its edges, the overall graph will seem more like many densely-connected vertex-disjoint subgraphs. The key to this strategy is a new lemma in our paper that allows for destroying large independent sets (equivalently, cliques) by adding "few" edges (equivalently, removing in the case of cliques).

1.3 Related Works

A similar-in-spirit result for vertex coloring is proven in [3], which shows that deterministic algorithms cannot color a graph with $\exp(\Delta^{o(1)})$ colors in a single pass. Since vertex coloring is fairly hard, designing an adversary entails finding *some* set of vertices that is a clique from the perspective of a deterministic algorithm. With the independent set problem, however, we must design an adversary that removes *all* large independent sets from the perspective of a deterministic algorithm. Due to this difference, deterministic algorithms can easily find independent sets of size $\frac{n}{\text{poly}(\Delta)}$ in our setting (whereas no deterministic algorithm can find a coloring using at most $\text{poly}(\Delta)$ colors). We show that they cannot do better than quadratic, up to a polylog factor.

More generally, there has also been a significant interest in finding independent sets in graph streams [2, 5, 15, 9, 10, 7, 6]. Independent sets in the *online streaming* model are studied in [16]. Under this model, they devise a deterministic algorithm with performance ratio $O(2^{\Delta})$, which they prove is also tight. We provide an adjacent result in the *semi-streaming* model, which does not require an algorithm to maintain a feasible solution at all times. Additionally, [12] studied independent sets in vertex-arrival streams, where each element in the input stream is a vertex along with its incident edges to earlier vertices. They show that the maximum independent set problem in the vertex-arrival model is not much easier than the problem in the edge-streaming model.

Independent sets have also been studied in more tangential settings. [11] studied the problem of finding the Caro-Wei bound itself that other algorithms (such as the one in [15]) achieve, and [7] studied the geometric independent set problem in the streaming model.

2 Technical Overview

In this section, we provide an informal sketch of our strategy. For clarity, we omit many details and technicalities.

As stated in Section 1.2, we prove Result 1 by considering the multi-party communication complexity of the independent set problem. Critical to our approach is the idea of non-edges: the set of all edges that a player *knows* has not been sent to any previous player based on their messages. The goal of our adversary is to ensure that the last player's graph of non-edges has small maximum clique size, which limits the largest independent set they can confidently output.

To do so, it will send a set of edges to each player sampled from a distribution that is chosen based on the behavior of previous players. As it progresses, the graph of non-edges will resemble a Turán graph. To achieve this, it will partition the vertex set into disjoint subsets. Within each subset, the adversary will send edges so that the non-edge graph induced by the subset is fairly sparse, resulting in a small clique number. Between subsets, there can be any number of edges, but the *overall* clique number is bounded above by the sum of the clique numbers of the non-edge subgraph induced by each subset.

2.1 The Adversary

Let n be the number of vertices, Δ the maximum degree, and s the memory size. Set the number of players k to be $\approx \log n$ and define $\ell = \Omega(\frac{s}{n})$. For the first player, the graph of non-edges is a clique on n vertices.

1. We first partition the graph of non-edges into $\frac{n}{\Delta^2}$ vertex-disjoint subgraphs of size Δ^2. Denote the union of these subgraphs as H. This is shown in Figure 1 (a).
2. For Player 1, sample each edge in H independently with probability $\frac{1}{\ell\Delta}$. Discard any sample that increases the degree of any vertex by $\geq \frac{2\Delta}{\ell}$. By the **Compression Lemma** (stated below), since each player is a compression algorithm, there exists a message M_1 such that Player 2's non-edge graph intersects H in at most $\widetilde{O}(s\ell\Delta)$ edges.

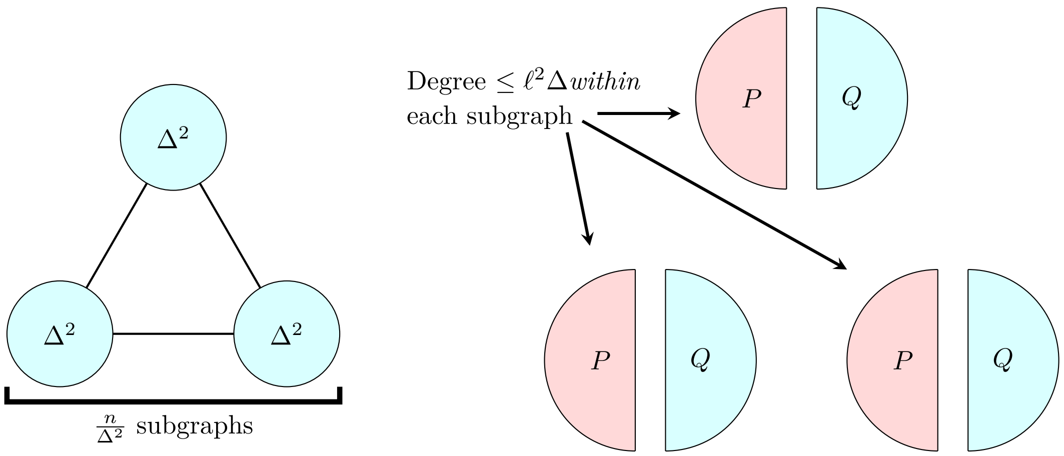
Δ^2
Δ^2
Δ^2
$\frac{n}{\Delta^2}$ subgraphs
Degree $\leq \ell^2\Delta$ *within* each subgraph
P
Q
P
Q
P
Q

Figure 1 (a) Partition the graph into vertex-disjoint subgraphs. (b) Further divide each subgraph based on non-edge degree.

3. Let H' be the graph containing these remaining non-edges. For each subgraph in H', split the vertices into P_j and Q_j, where P_j consists of vertices with degree $\leq \ell^2\Delta$ *within* the subgraph and Q_j consists of the remaining vertices. This is shown in Figure 1 (b).
   a. Since the degree of each vertex in Q_j (w.r.t. H') exceeds $\ell^2\Delta$ and there are $\lesssim s\ell\Delta$ edges in H', there are at most $O(\frac{s}{\ell})$ vertices in the union of all Q_j. We can combine these vertices into a single set Q of size $O(\frac{s}{\ell})$. This is shown in Figure 2.
   b. For each P_j, we can apply **Lemma 1** (stated below) and remove $\leq \frac{\Delta}{2}$ edges from each vertex in P_j (by sending those edges to Player 2) to ensure the maximum clique size in P_j is $\leq \widetilde{O}(\ell^2)$. This is shown in Figure 2.
4. Recurse on the non-edge graph induced on Q, which is now of size $O(\frac{s}{\ell}) < \frac{n}{c}$ for some $c > 1$ (we choose ℓ to be a large enough multiple of $\frac{s}{n}$). Repeat these same operations with Player 2. This time, we can partition the graph into $\frac{n}{\Delta^2}$ subgraphs with $\frac{\Delta^2}{c}$ vertices. Then, apply the **Compression Lemma** with probability $c \cdot \frac{1}{\ell\Delta}$ to bound the number of non-edges to $O\left(\frac{s\ell\Delta}{c}\right)$. This results in a new set of subgraphs P with clique number $\widetilde{O}(\ell^2)$ and a set Q of $< \frac{n}{c^2}$ vertices.
5. We repeat the process k times. This is enough for the set Q to have size $\leq \frac{n}{\Delta}^2$. At this stage, the graph of non-edges is made up of many vertex-disjoint subgraphs, each with degree $\widetilde{O}(\ell^2)$. The subgraphs are connected to each other through a (potentially) large number of edges. This is shown in Figure 3.
6. The largest clique in the last player's graph of non-edges has size at most the *sum* of the clique number of each subgraph, which is $\leq k \cdot \frac{n}{\Delta^2} \cdot \widetilde{O}(\ell^2) + \frac{n}{\Delta^2} = \widetilde{O}(\frac{n}{\Delta^2})$. This implies that the largest IS the last player can output has size $\widetilde{O}(\frac{n}{\Delta^2})$. Note that $\ell = O(\text{polylog}(n))$ in the semi-streaming model.

The compression lemma takes advantage of the bounded message size, allowing us to bound on the number of non-edges without significantly increasing the degree on each vertex.

Compression Lemma: Let H be any arbitrary graph with degree d. Consider the distribution of subgraphs obtained by sampling each edge in H with probability p while ignoring results that have degree $\geq 2pd$. For any compression algorithm that maps graphs sampled from this distribution into s-bit summaries, there is some summary such that at most $O(\frac{s}{p})$ edges are not in *any* graph mapped to this summary.

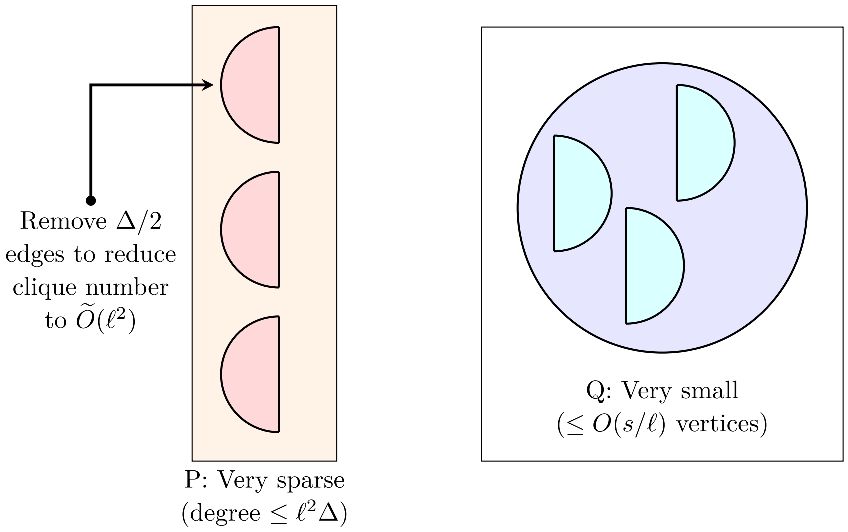
Remove $\Delta/2$ edges to reduce clique number to $\widetilde{O}(\ell^2)$
P: Very sparse (degree $\leq \ell^2\Delta$)
Q: Very small ($\leq O(s/\ell)$ vertices)

Figure 2 Combine the vertices with high degree into a single set.

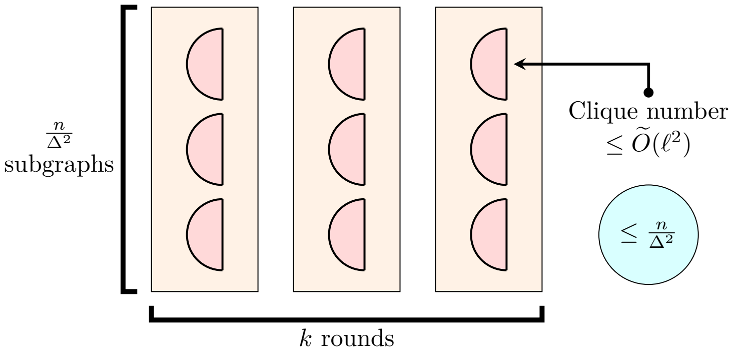
$\frac{n}{\Delta^2}$ subgraphs
Clique number $\leq \widetilde{O}(\ell^2)$
$\leq \frac{n}{\Delta^2}$
k rounds

Figure 3 The final graph of non-edges is made up of multiple vertex-disjoint subgraphs, each with a low clique number. The subgraphs are connected to each other through a (potentially) large number of edges.

The lemma below extends the compression lemma, allowing us to aggressively bound the maximum clique size in low-degree graphs without removing many edges from each vertex.

Lemma 1. Let G be an arbitrary graph with maximum degree Δ. For any positive integer d, there is some subgraph H of G with maximum degree $\leq d$ such that the largest clique size in $G - H$ has size $\widetilde{O}\left(\frac{\Delta}{d}\right)$.

3 Preliminaries

Notation. For an integer $k \geq 1$, we denote $[k] := \{1, 2, \ldots, k\}$. For a tuple $(X_1, \ldots, X_k)$ and any $i \in [k]$, we denote $X_{<i}$ as $(X_1, \ldots, X_{i-1})$. For any distribution μ, we will denote $\mathrm{supp}(\mu)$ as the support of μ.

For a graph $G = (V, E)$, we denote $\Delta(G)$ as the maximum degree of G, and for any $v \in V$, $\deg(v)$ as the degree of v in G. For any vertex set $T \subseteq V$, we will denote the induced subgraph of G on T as $G[T]$. Often, we will also partition a set S into a collection of subsets $\mathcal{P}$ and a subset Q. When we use this language, we are saying that $\mathcal{P} \cup \{Q\}$ is a partition of S.

One part of our strategy also involves partitioning the vertices into many small subsets. For any integer $g \geq 1$, we will denote $\textsc{Partition}(S, g)$ as an arbitrary partition of S into subsets of size g, *except* potentially for the last set, which has size $< g$.

▶ **Fact 1.** *For any set S and $g \geq 1$, $|\textsc{Partition}(S, g)| \leq \left\lceil \frac{|S|}{g} \right\rceil$.*

Finally, we use the following standard Chernoff bound:

▶ **Proposition 2** (Chernoff bound; c.f. [13]). *Suppose $X_1, \ldots, X_m$ are m independent random variables in the range $[0, 1]$. Let $X := \sum_{i=1}^{m} X_i$ and $\mu_L \leq \mathbb{E}[X] \leq \mu_H$. Then, for any $\epsilon > 0$,*

$$\Pr(X > (1 + \epsilon) \cdot \mu_H) \leq \exp\left(-\frac{\epsilon^2 \cdot \mu_H}{3 + \epsilon}\right) \text{ and } \Pr\left(X < (1 - \epsilon) \cdot \mu_L\right) \leq \exp\left(-\frac{\epsilon^2 \cdot \mu_L}{2 + \epsilon}\right).$$

3.1 The Communication Complexity of Independent Sets

We prove our space lower bound in Result 1 through a communication complexity argument in the following communication game, which as stated in Section 1.2, is defined similarly to [3].

▶ **Definition 3.** *For integers $n, \Delta, k, s \geq 1$, the $\textsc{Independent-Set}(n, \Delta, k, s)$ game is defined as:*

1. *There are k players $P_1, \ldots, P_k$. Each Player P_i knows the vertex set V and receives a set E_i of edges. Let $G = (V, E)$, where $E = E_1 \cup \cdots \cup E_k$ and players are guaranteed $E_1, \ldots, E_k$ are disjoint. Players are guaranteed $\Delta(G) \leq \Delta$, and their goal is to output an Independent set of G.*
2. *In order from $i = 1$ to $i = k$, each player P_i writes a public message M_i based on E_i and $M_{<i}$ (all the messages from the previous players) of length at most s.*
3. *The goal of the players is to output an independent set of G by P_k outputting it as the message M_k.*

The following is standard:

▶ **Lemma 4.** *Suppose there is a deterministic streaming algorithm that, on any n-vertex graph G with known maximum degree Δ, outputs an independent set of G with size r using s bits of space. Then, for any positive k, there also exists a deterministic protocol for $\textsc{Independent-Set}(n, \Delta, k, s)$ that outputs an independent set of size r.*

Proof. The players simply run the streaming algorithm on their input by writing the content of the memory of the algorithm from one player to the next on the blackboard, so that the next player can continue running the algorithm on their input. At the end, the last player computes the output of the streaming algorithm and writes it on the blackboard. ◁

3.2 The Missing Graph and Compression Lemma

As stated in Section 1.2, we use the compression lemma in [3]. We begin with two definitions that are similar to [3].

▶ **Definition 5.** *For a base graph $G_{Base} = (V, E_{Base})$ and parameters $p \in (0, 1]$, $d \geq 1$, we define the random graph distribution $\mathbb{G} = \mathbb{G}(G_{Base}, p, d)$ as follows:*

1. *Sample a graph G on vertices V and edges E by picking each edge of E_{Base} independently with probability p.*
2. *Return G if $\Delta(G) < 2p \cdot d$. Otherwise, repeat the process.*

To analyze arbitrary deterministic algorithms, we will often consider the compression algorithm associated with it. We will represent the "information" available to the algorithm as a "missing graph[2]", which we define here:

▶ **Definition 6.** *Consider $\mathbb{G}(G_{Base}, p, d)$ for a base graph $G_{Base} = (V, E_{Base})$ and parameters $p \in (0,1]$, $d \geq 1$, and an integer $s \geq 1$. A compression algorithm with size s is any function $\Phi : \text{supp}(\mathbb{G}) \to \{0,1\}^s$ that maps graphs sampled from $\mathbb{G}$ into s-bit strings. For any graph $G \in \text{supp}(\mathbb{G})$, we refer to $\Phi(G)$ as the summary of G. For any summary $\Phi \in \{0,1\}^s$, we define:*

1. *$\mathbb{G}_\phi$ as the distribution of graphs mapped to ϕ by Φ.*
2. *$G_{Miss}(\phi) = (V, E_{Miss}(\phi))$, called the missing graph of ϕ, as the graph on vertices V and edges missed by all graphs in $\mathbb{G}_\phi$. In other words,*

$$E_{Miss}(\phi) = \{(u,v) \in E_{Base} \mid \textit{no graph in } \text{supp}(G_\phi) \textit{ contains the edge } (u,v)\}$$

The previous definitions are used extensively. The following lemma also plays a crucial role in our communication lower bound, bounding the number of conclusive missing edges that can be recovered from a compression algorithm of a given size. It is proven in Lemma 4.3 of [3].

▶ **Lemma 7** (Compression Lemma). *Let $G_{Base} = (V, E_{Base})$ be an n-vertex graph, $s \geq 1$ be an integer, and $p \in (0,1)$ and $d \geq 1$ be parameters such that $d \geq \max\{\Delta(G_{Base}), \frac{4\ln(2n)}{p}\}$. Consider the distribution $\mathbb{G} := \mathbb{G}(G_{Base}, p, d)$ and suppose $\Phi : \text{supp}(\mathbb{G}) \to \{0,1\}^s$ is a compression algorithm of size s for $\mathbb{G}$. Then, there exists a summary $\phi^* \in \{0,1\}^s$ such that in the missing graph of ϕ^*,*

$$|E_{Miss}(\phi^*)| \leq \frac{\ln 2 \cdot (s+1)}{p}.$$

4 Removing Cliques in the Missing Graph

In this section, we introduce a key tool used by our adversary. More specifically, we develop a procedure that removes large cliques in the missing graph. This is useful because a deterministic algorithm that finds independent sets must be *certain* that none of the vertices in its output have an edge between them, which creates a clique of the same size in its graph of non-edges. To do so, we will design our adversary to generate a Turán-type graph, as mentioned in Section 1.2.

4.1 Removing Cliques in Low-Degree Graphs

The following lemma provides a method to bound the largest size of a clique when the degree is bounded by removing a small number of edges.

▶ **Lemma 8.** *Let G be a graph with n vertices such that $\Delta(G) \leq \Delta$. For any positive integer d, there is some subgraph H of G such that:*

1. *The degree of H is $\leq d$.*
2. *The largest clique in $G - H$ has size $\leq 16\ln(n) \cdot \frac{\Delta}{d} + 10$.*

[2] The missing graph is *not* the same as the graph of non-edges (which pertains to all previous players). However, they are closely related, and the missing graph is one component of the graph of non-edges.

Proof. We will use a probabilistic argument.

Firstly, if $d \leq 16\ln(n)$, then we let H be the empty graph. Since $\Delta(G) \leq \Delta$, the largest clique in G has size $\leq \Delta$. Indeed, $16\ln(n) \cdot \frac{\Delta}{d} + 10 \geq 16\ln(n) \cdot \frac{\Delta}{16\ln(n)} = \Delta$.

Similarly, if $\Delta \leq d$, then take $H = G$. The largest clique size is 1 and $\Delta(H) \leq \Delta \leq d$. Finally, if $n \leq 10$, then the largest clique size is at most 10, so we can let S be the empty graph as well.

It remains to prove the lemma for $n > 10$ and $16\ln(n) < d < \Delta$. In this case, H is a random subgraph chosen by sampling each edge in G with probability $\frac{d}{2\Delta}$. For any vertex v, we let $X_1, \ldots, X_{\deg(v)}$ be indicator variables for whether an incident edge is chosen. If we let $X := \sum_{i=1}^{\deg(v)} X_i$, then $\mathbb{E}[X] \leq \frac{d}{2}$. By Proposition 2 with $\epsilon = 1$,

$$\Pr(X > d) \leq \Pr\left(X > 2 \cdot \mathbb{E}[X]\right) \leq \exp\left(-\frac{\frac{d}{2}}{4}\right) < \exp\left(-\frac{16\ln n}{8}\right) = \exp(-2\ln n).$$

Hence, by a union bound, Pr(Some vertex in H has degree greater than d) does not exceed

$$n \cdot \exp\left(-2\ln n\right) = \exp(\ln n - 2\ln n) = \exp(-\ln n) = \frac{1}{n}.$$

For any group of $16\ln(n) \cdot \frac{\Delta}{d} + 10 < k$ vertices, if it is not a clique in G, it definitely will not be a clique after removing the edges in H. Otherwise, it can only be a clique if none of the edges are selected. For any k-subset, the probability of this happening is:

$$\begin{aligned}\Pr(\text{this } k\text{-subset is a clique}) &= \left(1 - \frac{d}{2\Delta}\right)^{k\cdot(k-1)/2} \\ &\leq \exp\left(-\frac{d}{2\Delta} \cdot \frac{k(k-1)}{2}\right) \\ &< \exp\left(-\frac{d}{2\Delta} \cdot \left(16\ln n \cdot \frac{\Delta}{d}\right) \cdot \frac{k-1}{2}\right) \\ &= \exp\left(-4\ln n \cdot (k-1)\right).\end{aligned}$$

Now, we apply a union bound over all k-subsets of vertices:

$$\begin{aligned}\Pr(G - H \text{ has a k-clique}) &\leq \binom{n}{k} \cdot \exp\left(-4\ln n \cdot (k-1)\right) \\ &\leq \left(\frac{en}{k}\right)^k \cdot \exp\left(-4\ln n \cdot (k-1)\right) \\ &= \exp(k\ln(en/k) - 4\ln(n) \cdot (k-1)) \\ &= \exp(k(1 + \ln(n) - \ln(k) - 4\ln(n)) + 4\ln n) \\ &\leq \exp(k(1 - 3\ln(n)) + 4\ln(n)) \\ &\leq \exp(-2k\ln n + 4\ln(n)) && \text{(Since } \ln(n) \geq 1) \\ &\leq \exp\left(-6\ln(n) + 4\ln(n)\right) \\ & && \text{(Sub } k = 16\ln(n) \cdot \tfrac{\Delta}{d} \geq 3) \\ &= \frac{1}{n^2}.\end{aligned}$$

Finally, we calculate the probability that this procedure yields a subgraph H satisfying Item 1 and Item 2. Through a union bound on the complement, when $n > 10$, we get

$$\Pr(H \text{ does not satisfy Item 1 or Item 2}) \leq \frac{1}{10} + \frac{1}{100} < 1.$$

Thus, there is some H satisfying the two conditions. ◁

4.2 Removing Cliques in General Graphs

To use Lemma 8, we need graphs of low maximum degree. However, the Compression Lemma can only bound the total number of edges. Hence, to employ Lemma 8, we will partition a graph with few edges into many subgraphs with low degree and a small "remainder" subgraph. We will start with the following definition:

▶ **Definition 9.** *For a positive integer b and a graph $G = (V, E)$ with m edges, we define* $\textsc{Split}(G, b)$ *as a partition of V into a pair of vertex sets (P, Q) such that $\Delta(G[P]) \leq b$ and $|Q| \leq \frac{2m}{b}$.*

A split necessarily exists for regular graphs. In the general case, the following proposition ensures that a split exists:

▶ **Proposition 10.** *Suppose we are given a graph $G = (V, E)$ with n vertices and m edges. For every positive integer b,* $\textsc{Split}(G, b)$ *exists.*

Proof. Let $P := \{v \in V : \deg(v) \leq b\}$. Then, $Q := V \backslash P$. This is clearly a partition since each vertex is in exactly one of P or Q.

We will now prove the bound on $G[P]$. For every vertex $v \in P$, $\deg(v) \leq b$ in G. Since $G[P]$ is a subgraph of G, $\deg(v) \leq b$ in $G[P]$ as well. Hence, $\Delta(G[P]) \leq b$.

Now for the bound on $|Q|$. It is known that the number of edges is equal to half the sum of the degrees. Hence, we have

$$m = \frac{1}{2}\sum_{v \in V} \deg(v) \geq \frac{1}{2}\sum_{v \in V \backslash P} \deg(v) \geq \frac{1}{2}\sum_{v \in Q} b = \frac{1}{2} \cdot |Q| \cdot b.$$

Then, we rearrange to get $|Q| \leq \frac{2m}{b}$. ◁

To "remove" many large cliques in an arbitrary graph, we will run a two-step subalgorithm.

The first step of the subalgorithm involves finding a subgraph of our input graph G_{Base}, which we denote as H_{Base}. The graph H_{Base} is chosen such that, for any algorithm compressing it, we can easily bound the number of edges in $H_{Miss}(\phi)$ (the missing graph of H_{Base} for some message ϕ) using the Compression Lemma. We will then prove that we can partition the vertices of $H_{Miss}(\phi)$ (which are the same as the vertices of G_{Base}) into a collection of vertex sets $\mathcal{P}$ and a vertex set Q such that the maximum degree in $(H_{Miss}(\phi))[P]$ is small for all $P \in \mathcal{P}$ and the size of Q is small.

▶ **Lemma 11.** *Let $G_{Base} = (V, E_{Base})$ be a graph with n vertices. Let $g \geq 1$ (group size) and $s \geq 1$ (message size) be arbitrary integers. Let $d_{comp} \geq 4\ln(2n)$ (compression degree) and $d_{filter} \geq 1$ (filter degree) be arbitrary real numbers.*

Then, there is a subgraph $H_{Base} \subseteq G_{Base}$ and a distribution $\mathbb{H} := \mathbb{G}(H_{Base}, p, d)$ (for some p and d) such that, for every compression algorithm $\Phi : \mathrm{supp}(\mathbb{H}) \to \{0,1\}^s$, we can find a message ϕ and a partition of V into a collection of vertex sets $\mathcal{P}$ and a vertex set Q satisfying:

1. *For all $H \in \mathrm{supp}(\mathbb{H})$, $\Delta(H) \leq 2 \cdot d_{comp}$.*
2. *For all $P \in \mathcal{P}$, $G_{Base}[P] = H_{Base}[P]$.*
3. *For any vertex set $P \in \mathcal{P}$, $\Delta(H_{Miss}(\phi)[P]) \leq d_{filter}$.*
4. *The size of $\mathcal{P}$ is $\leq \lceil \frac{n}{g} \rceil$.*
5. *The size of Q is $\leq \dfrac{2\ln(2) \cdot (s+1) \cdot g}{d_{comp} \cdot d_{filter}}$.*

Proof. Let $H := \textsc{Partition}(V, g)$ and $H_{Base} := \bigcup_{S \in H} G_{Base}[S]$. We prove this lemma through casework on d_{comp}.

If $d_{comp} < g$, then we define $\mathbb{H} := \mathbb{G}\left(H_{Base}, \frac{d_{comp}}{g}, g\right)$.

For all $K \in \text{supp}(\mathbb{H})$, $\Delta(K) \le 2 \cdot \frac{d_{comp}}{g} \cdot g = 2 \cdot d_{comp}$ by construction, proving Item 1.

These values of $p = \frac{d_{comp}}{g}$ and $d = g$ satisfy the requirements for the Compression Lemma. In particular, since $4\ln(2n) \le d_{comp} < g$, we have $p = \frac{d_{comp}}{g} \in (0, 1)$. Additionally, $d = g \ge \Delta(H_{Base})$ since each subgraph has at most g vertices, and $d = g = \frac{d_{comp}}{d_{comp}/g} \ge \frac{4\ln(2n)}{p}$. Hence, the values of p and d satisfy the necessary constraints.

Therefore, there exists some message ϕ such that $H_{Miss}(\phi) = (V, E_{Miss}(\phi))$ satisfies

$$|E_{Miss}(\phi)| \le \frac{\ln(2) \cdot (s+1) \cdot g}{d_{comp}}.$$

Let $(L, Q) := \textsc{Split}(H_{Miss}(\phi), d_{filter})$. By Proposition 10, $|Q| \le \frac{2\ln(2)\cdot(s+1)g}{d_{comp}\cdot d_{filter}}$, proving Item 5. Additionally, let $\mathcal{P} := \{S \cap L : S \in H\}$. Since H is a partition of V and $Q = V \backslash L$, we know that $\mathcal{P} \cup \{Q\}$ is a partition of V. By Fact 1, $|\mathcal{P}| = |H| \le \lceil \frac{n}{g} \rceil$, proving Item 4.

We will now prove the conditions on all $P \in \mathcal{P}$. Let $X_P := H_{Miss}(\phi)[P]$.

- By Proposition 10, since $P \subseteq L$, $\Delta(X_P) \le \Delta(H_{Miss}(\phi)[L]) \le d_{filter}$. This proves Item 3.
- We know $P \subseteq S$ for some $S \in H$. Additionally, H is a partition of V. So, by our construction of H_{Base}, we have $H_{Base}[S] = G_{Base}[S]$. Hence, since $P \subseteq S$, we have $H_{Base}[P] = G_{Base}[P]$. This proves Item 2.

If $d_{comp} \ge g$, then we let $\mathbb{H} = \Phi(H_{Base}, 1, |V|)$. We let $\mathcal{P} = H$, so $Q = \emptyset$, which satisfies Item 5. The distribution $\mathbb{H}$ is a single graph H_{Base}. Since each subgraph in H_{Base} has $\le g$ vertices, $\Delta(H_{Base}) \le g \le 2 \cdot d_{comp}$, proving Item 1. Since $|\mathcal{P}| = |H|$, Item 4 is proven by Fact 1. Since $\text{supp}(\mathbb{H}) = \{H_{Base}\}$, $H_{Miss}(\phi) = \emptyset$, proving Item 3. By how we defined $\mathcal{P}$ and H_{Base} (note that H is a partition), $G_{Base}[P] = H_{Base}[P]$ for all $P \in \mathcal{P}$. This proves Item 2. ◁

In the second step of the subalgorithm, we will apply Lemma 8 on $H_{Miss}(\phi)[P]$ for all $P \in \mathcal{P}$, storing the removed edges in a subgraph R (which will have low degree). In the end, the maximum clique size in $H_{Miss}(\phi) - R$ will be small since it cannot exceed the *sum* of the maximum clique sizes over the subgraphs induced by the vertex sets in $\mathcal{P}$.

▶ **Lemma 12.** *Suppose we have a graph H with n vertices and a partition of its vertices into a collection of vertex sets $\mathcal{P}$ and a vertex set Q. Additionally, for all $P \in \mathcal{P}$, suppose that the degree of $G[P]$ does not exceed d_{filter}.*

Then, for any integer $d_{remove} > 0$ (removal degree) there is a subgraph $R \subseteq H$ such that:

- *For all $P \in \mathcal{P}$, the largest clique in $(H_{Miss}(\phi) - R)[P]$ has size $\le 16\ln(n) \cdot \frac{d_{filter}}{d_{remove}} + 10$.*
- *None of the edges in R is incident to a vertex in Q.*
- *The degree of R does not exceed d_{remove}.*

Proof. For all $P \in \mathcal{P}$, by Lemma 8 on $H[P]$ and d_{remove}, there is a graph R_P with degree $\le d_{remove}$ s.t. the maximum clique size in $H[P] - R_P$ is $\le 16\ln(n) \cdot \frac{d_{filter}}{d_{remove}} + 10$.

We take $R := \bigcup_{P \in \mathcal{P}} R_P$. Since Q is disjoint with all sets in $\mathcal{P}$, none of the edges in R are incident to any vertex in Q. Additionally, for all $P \in \mathcal{P}$, since $R_P \subseteq R$ and the vertex sets in $\mathcal{P}$ are disjoint, we have $R_P = R[P]$. Thus, the largest clique in $(H - R)[P] = H[P] - R_P$ has size $\le 16\ln(n) \cdot \frac{d_{filter}}{d_{remove}} + 10$. ◁

5 A Communication Lower Bound for Independent Set

We will prove our lower bound in Result 1 by showing that it is impossible for any set of players to output a large independent set, which is sufficient by Lemma 4. To do so, we will design an adversary that, for any large independent set, can find an *invalid* graph and a set of edges to send to each player such that each player outputs the same message. This will ensure that no deterministic algorithm can confidently output any large independent set.

5.1 The Adversary

▶ **Theorem 13.** *There exist constants $\eta > 0$ and $\eta_0 > 0$ such that: if n, s, and Δ are integers satisfying*

$$\eta < n \le s, \quad \max\left\{\eta_0 \cdot \frac{s\ln(n)}{n}, \eta_0 \cdot (\ln n)^2\right\} < \Delta < \sqrt{n},$$

then the size of the largest independent set a deterministic algorithm using s bits of memory can output for all graphs of size n and maximum degree Δ does not exceed $\widetilde{O}\left(\frac{s}{\Delta^2} \cdot \frac{s}{n}\right)$.

Under the semi-streaming model, $s = \widetilde{O}(n)$. Consequently, the largest independent set that a deterministic algorithm can find under this model has size $\widetilde{O}\left(\frac{n}{\Delta^2}\right)$, which formalizes Result 1.

To begin with the proof of Theorem 13, we will let $\eta = e^2$ and $\eta_0 = 128$. We let $\ell = \max\left\{\left\lceil \frac{2e\ln(2)(s+1)}{n}\right\rceil, \lceil 8\ln n\rceil\right\}$. By our choice of η and η_0, it is easy to prove that $\ell < \frac{\Delta}{4\ln(2n)}$. We let $k = \lceil \ln n\rceil + 1$. For our adversary, k denotes the number of players, and we will generally send graphs of degree $\le \frac{\Delta}{\ell}$ to each player.

At a high level, our adversary adheres to the following structure:

- For all $i = 1 \ldots k$, $G_{Base}(i)$ is the graph where most edges for Player i will be chosen from.
- Similarly, R_i will be provided as an additional set of edges to send to Player i to "manually" remove large cliques in the missing graph for the previous players' messages.
- For all $i \in [k]$, Player i receives R_i and a subgraph $H_i \subseteq G_{Base}(i)$.
- Lemma 11 is used to find an adversarial distribution of subgraphs $\mathbb{H}_i$, which is used to obtain H_i. Lemma 12 is used to determine R_{i+1}, which is a subgraph of $G_{Base}(i)$.
- *All* graph parameters are derived adversarily based on what each player communicates. The compression algorithm associated with each player also affects the input for future players by determining $G_{Base}(i+1)$ and R_{i+1}.
- The adversarily generated input graph, G_{input}, is the union of all graphs H_i and R_i.

An adversary that generates a "hard" input

1. We start with $G_{Base}(1)$ as the clique on n vertices and R_1 as an empty graph.
2. For $i = 1 \ldots k$,
   a. Let n_i be the number of vertices in $G_{Base}(i)$, where $G_{Base}(i) = (V_{Base}(i), E_{Base}(i))$.
   b. If $n_i < \frac{n}{\Delta^2}$, then terminate the algorithm, letting $\mathbb{H} = \emptyset$ and $R_{i+1} = \emptyset$. For the sake of our analysis, we let $G_{Base}(i+1) = G_{Base}(i)$ and run the adversary until $i = k$.
   c. Otherwise, apply Lemma 11 with $G_{Base}(i)$, group size $\lfloor \frac{n_i\Delta^2}{n} \rfloor$, message size s, compression degree $\frac{\Delta}{\ell}$, and filter degree $\ell^2\Delta$.
   We have a distribution $\mathbb{H}_i$ with a base graph $H_{Base}(i)$, and we define $\Phi_i = \Phi(\mathbb{H}_i, M_{<i})$ as the compression algorithm for Player i *after* receiving R_i.
   By Lemma 11, there is a message $M_i := \phi$ and some partition of $V_{Base}(i)$ into a collection of vertex sets $\mathcal{P}_i$ and a vertex set Q_i satisfying the conclusions of Lemma 11.

  - **d.** Apply Lemma 12 with $H_{Miss}(M_i)$ and removal degree $\frac{\Delta}{2}$ to find some R_{i+1} such that the conclusions of Lemma 12 holds.
  - **e.** Let $G_{Base}(i+1) = (G_{Base}(i) - (H_{Base}(i) - H_{Miss}(M_i)))[Q_i]$.

3. Finally, we generate the edges we send each player. For $i = 1, \dots, k$,
  - **a.** If $\mathbb{H} = \emptyset$, let $H_i = \emptyset$. Otherwise, choose $H_i \in \text{supp}(\mathbb{H}_i)$ such that $\Phi_i(H_i) = M_i$.
  - **b.** We send Player i the graph $H_i \cup R_i$.

Then, the input graph is the union of these graphs. In particular, $G_{input} := \bigcup_{i=1}^{k} H_i \cup R_i$.

To begin, we must ensure that the input graph G_{input} is valid. The following lemma ensures that we are not sending a multigraph to the players.

▶ **Lemma 14.** *No two players will receive the same edge (i.e. G_{input} is not a multigraph).*

Proof. It is sufficient to prove the stronger statement that $\{H_i : i \in [k]\} \cup \{R_{i+1} : i \in [k]\}$ is disjoint. For each $i \in [k]$, by Lemma 11 and Lemma 12, $H_{Base}(i) \subseteq G_{Base}(i)$ and $R_{i+1} \subseteq H_{Miss}(M_i) \subseteq H_{Base}(i) \subseteq G_{Base}(i)$. Hence, both H_i and R_{i+1} are subgraphs of $G_{Base}(i)$.

Firstly, we claim that H_i and R_{i+1} are edge-disjoint. In particular, since H_i is sampled from $\text{supp}(\mathbb{H}_i)$ and $\Phi_i(H_i) = M_i$, by definition, we know that none of the edges in H_i are in $H_{Miss}(M_i)$. However, $R_{i+1} \subseteq H_{Miss}(M_i)$, so none of the edges in R_{i+1} are in H_i.

Next, we prove that $G_{Base}(i+1)$ is edge-disjoint with H_i and R_{i+1}. For any edge in H_i, it is in $H_{Base}(i)$ but not $H_{Miss}(i)$. Hence, it is not in $G_{Base}(i) - (H_{Base}(i) - H_{Miss}(M_i))$. For R_{i+1}, by Lemma 12, none of its edges are incident to a vertex in Q_i. Thus, none of the edges in R_{i+1} appear in $G_{Base}(i+1) = (G_{Base}(i) - (H_{Base}(i) - H_{Miss}(M_i)))[Q_i]$.

Finally, for any $j > i$, we know that $G_{Base}(j) \subseteq \cdots \subseteq G_{Base}(i+1)$. Thus, for any edge (u,v) in H_i or R_{i+1}, we know that (u,v) is not in $G_{Base}(j)$ (since $(u,v) \notin G_{Base}(i+1)$). Since H_j and R_{j+1} are subgraphs of $G_{Base}(j)$, (u,v) will not be in H_j or R_{j+1} either.

Thus, an edge in H_i or R_{i+1} (it only appears in one of the two) will not be sent in any H_j or R_{j+1} if $j > i$. As such, $\{H_i : i \in [k]\} \cup \{R_{i+1} : i \in [k]\}$ is edge-disjoint. Since R_1 is empty, this set contains all the graphs sent to any player. Hence, no two players will receive the same edge. ◁

Next, we must also ensure that the maximum degree of the input graph agrees with the maximum degree given to each player.

▶ **Lemma 15.** *For each vertex $v \in G_{input}$, $\deg(v) \le \Delta$.*

Proof. We consider the degree on each vertex v. Firstly, there is at most one value of i for which v is incident to some edge in R_i. After all, by Lemma 12, if an edge in R_i is incident to v, then $v \notin Q_{i-1}$ and v is not in $G_{Base}(i)$ nor any subsequent graphs. For this value of i, $\Delta(R_i) \le \frac{\Delta}{2}$. Furthermore, $\Delta(H_j) \le \frac{2\Delta}{\ell}$ for all $j \in [k]$ by Lemma 11. Thus, $\deg(v)$ cannot exceed $\frac{\Delta}{2}$ in R_i nor $\frac{2\Delta}{\ell}$ in H_j for any $j \in [k]$.

Since $G_{input} = \left(\bigcup_{j=1}^{k} H_j\right) \cup \left(\bigcup_{i=1}^{k} R_i\right)$, the degree of v in G_{input} is the sum of its degree in each of these graphs. In particular, the degree of v is $\le k \cdot \frac{2\Delta}{\ell} + \frac{\Delta}{2} \le \frac{4\ln(n)\cdot\Delta}{\ell} + \frac{\Delta}{2} \le \frac{\Delta}{2} + \frac{\Delta}{2} = \Delta$. The second last inequality is true by $k = \lceil \ln(n) \rceil + 1$ and $n > \eta$. The last is true by $\ell \ge 8 \ln n$. ◁

Lemma 14 and Lemma 15 prove that the input graph is valid. The next step is to prove that the adversary terminates – for the sake of our analysis, we will instead show that the final base graph $G_{Base}(k)$ is small.

▶ **Lemma 16.** $n_k \leq \frac{n}{\Delta^2}$.

Proof. We begin by proving that $n_i \leq \max(\frac{n}{\Delta^2}, \frac{n}{e^{i-1}})$, which we will show through induction.

For the base case, $n_i = n \leq \frac{n}{e^0}$.

For the inductive step, if $n_i \leq \frac{n}{\Delta^2}$, then $n_{i+1} \leq \frac{n}{\Delta^2}$ too. Otherwise, by Lemma 11,

$$|Q_i| \leq \frac{2\ln(2) \cdot (s+1) \cdot \lfloor \frac{n_i\Delta^2}{n} \rfloor}{\frac{\Delta}{\ell} \cdot \ell^2\Delta}.$$

Dropping the floor, the above is $\leq \frac{2\ln(2)\cdot(s+1)\cdot n_i\Delta^2}{n\ell\Delta^2} = \frac{2\ln(2)\cdot(s+1)}{n\ell} \cdot n_i \leq \frac{1}{e} \cdot \frac{n}{e^{i-1}} = \frac{n}{e^i}$.

The second last inequality is true by the bound on ℓ, while the last is true by the inductive hypothesis. Note that $n_{i+1} = |Q_i|$. This concludes the inductive step.

To prove Lemma 16, we note that $\frac{n}{e^{k-1}} \leq \frac{n}{e^{\ln n}} \leq 1$. Since $\Delta < \sqrt{n}$, we have $\frac{n}{\Delta^2} > 1 > \frac{n}{e^{k-1}}$, so $\max\left(\frac{n}{\Delta^2}, \frac{n}{e^{k-1}}\right) = \frac{n}{\Delta^2}$. We showed that $n_k \leq \max\left(\frac{n}{\Delta^2}, \frac{n}{e^{k-1}}\right)$, so $n_k \leq \frac{n}{\Delta^2}$. ◁

Having proven that the input graph is valid and the algorithm terminates, it remains to bound the largest clique at each iteration. Lemma 12 allows us to bound the largest clique in $H_{Miss}(\phi)[P]$ for all $P \in \mathcal{P}_i$. The following lemma bounds the size of $\mathcal{P}_i$, which will allow us to bound the size of the largest clique in the missing graph over the *entirety* of $\mathcal{P}_i$.

▶ **Lemma 17.** *For all $i \in [k]$, the size of $\mathcal{P}_i$ is $\leq \frac{3n}{\Delta^2}$.*

Proof. By Lemma 11, $|\mathcal{P}_i| \leq \left\lceil \frac{n_i}{\lfloor \frac{n_i\Delta^2}{n} \rfloor} \right\rceil \leq \left\lceil \frac{n_i}{\frac{n_i\Delta^2}{2n}} \right\rceil \leq \lceil \frac{2n}{\Delta^2} \rceil \leq \frac{2n}{\Delta^2} + 1 \leq \frac{3n}{\Delta^2}$.

The second inequality is true because $\frac{n_i\Delta^2}{n} \geq 1$. The last inequality is true by $\Delta < \sqrt{n}$. ◁

Finally, we prove that Player k cannot conclusively find a large independent set by proving that the adversary can find a "breaking" graph if the output is ever too large.

▶ **Lemma 18.** *Suppose Player k outputs an Independent set A of size greater than*

$$\frac{n}{\Delta^2} + \frac{n}{\Delta^2} \cdot k \cdot (96\ell^2 \ln(n) + 30).$$

Then, there is another graph G'_{input} and set of edges to send to each player such that each player outputs the same message but an edge exists between some u, v in the output of Player k.

Proof. For each $v \in A$, either $v \in G_{Base}(k+1)$ or $v \in P$ for some $i \in [k]$ and $P \in \mathcal{P}_i$. We let $L_i := \bigcup_{P\in\mathcal{P}_i} P$. Since $n_k < \frac{n}{\Delta^2}$, more than $\frac{nk(96\ell^2\ln(n)+30)}{\Delta^2}$ vertices in A are also in $L_1 \cup \cdots \cup L_k$. By the Pigeonhole principle, for some $i \in [k]$, we have $|L_i \cap A| > \frac{n(96\ell^2\ln(n)+30)}{\Delta^2}$.

Since $\mathcal{P}_i$ is a partition of the vertices in L_i, and since $|\mathcal{P}_i| \leq \frac{3n}{\Delta^2}$ by Lemma 17, there is some $P = (V_P, E_P) \in \mathcal{P}_i$ such that $|P \cap A| > 32\ell^2\ln(n) + 10$ (again, by the Pigeonhole principle).

By Lemma 11 and Lemma 12, the largest clique in $(H_{Miss}(M_i) - R_{i+1})[P]$ has size at most

$$16\ln(n) \cdot \frac{d_{filter}}{d_{removal}} + 10 = 16 \cdot \frac{\ell^2\Delta}{\Delta/2} \cdot \ln(n) + 10 = 32\ell^2\ln(n) + 10.$$

Thus, there is some $u, v \in P$ such that $(u, v) \notin H_{Miss}(M_i) - R_{i+1}$.

If $(u,v) \in R_{i+1}$, then we send the same graph and edges. Note that $R_{k+1} = \emptyset$ since $n_k < \frac{n}{\Delta}^2$. Thus, Player $i+1$ exists and will receive R_{i+1}. As such, there is an edge between (u,v) in G_{input}.

Otherwise, we know there is some $j \in [i]$ s.t. $(u,v) \in H_{Base}(j)$ but $(u,v) \notin H_{Miss}(M_j)$. We prove this with two cases:

If $(u,v) \in G_{Base}(i)$, then let $j = i$. Since $H_{Base}(i)[P] = G_{Base}(i)[P]$, we have $(u,v) \in H_{Base}(i)$. Additionally, since $(u,v) \notin R_{i+1}$, we know that $(u,v) \notin H_{Miss}(M_j)$ as well.

Otherwise, $(u,v) \notin G_{Base}(i)$. Since $G_{Base}(i) \subseteq \cdots \subseteq G_{Base}(1)$ and $(u,v) \in G_{Base}(1)$, we can find some $j < i$ s.t. $(u,v) \in G_{Base}(j)$ but $(u,v) \notin G_{Base}(j+1)$. In particular,

$$(u,v) \notin G_{Base}(j+1) = \big(G_{Base}(j) - (H_{Base}(j) - H_{Miss}(M_j))\big)[Q_j].$$

Since $u,v \in G_{Base}(i)$ and all vertices in $G_{Base}(i)$ are also in Q_j, we know that $u,v \in Q_j$. This allows us to conclude that

$$(u,v) \notin G_{Base}(j) - (H_{Base}(j) - H_{Miss}(M_j)).$$

We know that $(u,v) \in G_{Base}(j)$, so we must also have $(u,v) \in H_{Base}(j) - H_{Miss}(M_j)$. Hence, $(u,v) \in H_{Base}(j)$ and $(u,v) \notin H_{Miss}(M_j)$.

Since $(u,v) \notin H_{Miss}(j)$, and $(u,v) \in H_{Base}(j)$, there is some graph $H' \in \mathbb{H}_i$ such that $(u,v) \in H'$ and $\Phi_i(H') = M_i$. We substitute H_i with H' when choosing G_{input} and the edges we send each player. Each player sends the same message, but $(u,v) \in G'_{input}$, concluding the proof. ◁

Lemma 18 leads naturally to a bound on the largest independent set a deterministic algorithm can find, which the following formalizes.

▶ **Lemma 19.** *There does not exist a deterministic algorithm using at most s bits of memory that outputs an independent set of size greater than $\frac{n}{\Delta^2} + \frac{n}{\Delta^2} \cdot k \cdot (96\ell^2 \ln(n) + 30)$ for all graphs with n vertices and maximum degree Δ.*

Proof. Suppose a deterministic algorithm exists. Then, by Lemma 4, there is a set of $k = \lceil \ln n \rceil + 1$ players that can solve the INDEPENDENT-SET(n, Δ, k, s) problem with the same independent set size. However, by Lemma 18, this is impossible. After all, if Player k outputs an independent set of size greater than $\frac{n}{\Delta^2} + \frac{n}{\Delta^2} \cdot k \cdot (96\ell^2 \ln(n) + 30)$, then we can find another set of edges to send such that each player sends the same message but the independent set is no longer valid. ◁

To conclude, we substitute $\ell = \max\left\{\left\lceil \frac{2e\ln(2)(s+1)}{n} \right\rceil, \lceil 8 \ln n \rceil\right\}$ and $k = \lceil \ln n \rceil + 1$ into the bounds we showed in Lemma 19.

Proof of Theorem 13. This proof is simply a matter of loosening the bound in Lemma 19 until we get a "simpler" form. We will start with the choices of k and ℓ. For k, we have

$$k = \lceil \ln n \rceil + 1 \le 2 \ln n + 1 \le 3 \ln n.$$

For the first possible value of ℓ, we have

$$\left\lceil \frac{2e\ln(2)(s+1)}{n} \right\rceil \le \left\lceil \frac{4e\ln(2) \cdot s}{n} \right\rceil \le 8e\ln(2) \cdot \frac{s}{n}.$$

For the other potential value of ℓ, we have

$$\lceil 8 \ln n \rceil \le 16 \ln n.$$

Hence, $\ell \leq \max\{8e\ln 2 \cdot \frac{s}{n}, 16\ln n\}$. Since both terms are positive and greater than 1, we can multiply both upper bounds to get

$$\ell \leq 8e\ln 2 \cdot \frac{s}{n} \cdot 16\ln n = 128e\ln 2 \cdot \frac{s\ln n}{n}.$$

To simplify calculations, we will let $r = 128e\ln(2)$, which is a constant.

By Lemma 19, no deterministic algorithm can find an independent set of size greater than

$$\begin{aligned}\frac{n}{\Delta^2} + \frac{n}{\Delta^2} \cdot k \cdot (96\ell^2\ln(n) + 30) &\leq \frac{n}{\Delta^2}\left(1 + 3\ln(n) \cdot \left[96 \cdot \left(\frac{rs\ln n}{n}\right)^2 \cdot \ln(n) + 30\right]\right) \\ &\leq \frac{n}{\Delta^2} \cdot 2 \cdot 6(\ln(n))^2\left(96 \cdot \left(\frac{rs\ln n}{n}\right)^2\right) \\ &= 1152r^2\ln^4(n) \cdot \frac{s^2}{n\Delta^2} \\ &= \widetilde{O}\left(\frac{s}{\Delta^2} \cdot \frac{s}{n}\right).\end{aligned}$$

Hence, the size of the largest independent set that a deterministic algorithm can find for *all* graphs with n vertices and maximum degree Δ does not exceed $\widetilde{O}\left(\frac{s}{\Delta^2} \cdot \frac{s}{n}\right)$. ◀

References

1 Kook Jin Ahn, Sudipto Guha, and Andrew McGregor. Analyzing graph structure via linear measurements. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 459–467. SIAM, 2012. doi:10.1137/1.9781611973099.40.

2 KookJin Ahn, Graham Cormode, Sudipto Guha, Andrew McGregor, and Anthony Wirth. Correlation clustering in data streams. In *International Conference on Machine Learning*, pages 2237–2246. PMLR, 2015.

3 Sepehr Assadi, Andrew Chen, and Glenn Sun. Deterministic graph coloring in the streaming model. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 261–274, 2022. doi:10.1145/3519935.3520016.

4 Sepehr Assadi, Yu Chen, and Sanjeev Khanna. Sublinear algorithms for $(\Delta+1)$ vertex coloring. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 767–786. SIAM, 2019. doi:10.1137/1.9781611975482.48.

5 Sepehr Assadi, Christian Konrad, Kheeran K Naidu, and Janani Sundaresan. $O(\log\log n)$ passes is optimal for semi-streaming maximal independent set. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 847–858, 2024. doi:10.1145/3618260.3649763.

6 Ainesh Bakshi, Nadiia Chepurko, and David P. Woodruff. Weighted maximum independent set of geometric objects in turnstile streams. In Jaroslaw Byrka and Raghu Meka, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2020, August 17-19, 2020, Virtual Conference*, volume 176 of *LIPIcs*, pages 64:1–64:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPICS.APPROX/RANDOM.2020.64.

7 Sujoy Bhore, Fabian Klute, and Jelle J Oostveen. On streaming algorithms for geometric independent set and clique. In *International Workshop on Approximation and Online Algorithms*, pages 211–224. Springer, 2022. doi:10.1007/978-3-031-18367-6_11.

8 Vladimir Braverman, Rafail Ostrovsky, and Dan Vilenchik. How hard is counting triangles in the streaming model? In *Automata, Languages, and Programming: 40th International Colloquium, ICALP 2013, Riga, Latvia, July 8-12, 2013, Proceedings, Part I 40*, pages 244–254. Springer, 2013. doi:10.1007/978-3-642-39206-1_21.

9 Xiuge Chen, Rajesh Chitnis, Patrick Eades, and Anthony Wirth. Sublinear-space streaming algorithms for estimating graph parameters on sparse graphs. In *Algorithms and Data Structures Symposium*, pages 247–261. Springer, 2023. doi:10.1007/978-3-031-38906-1_17.

10 Graham Cormode, Jacques Dark, and Christian Konrad. Independent set size approximation in graph streams. *arXiv preprint arXiv:1702.08299*, 2017. arXiv:1702.08299.

11 Graham Cormode, Jacques Dark, and Christian Konrad. Approximating the caro-wei bound for independent sets in graph streams. In *International Symposium on Combinatorial Optimization*, pages 101–114. Springer, 2018. doi:10.1007/978-3-319-96151-4_9.

12 Graham Cormode, Jacques Dark, and Christian Konrad. Independent sets in vertex-arrival streams. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 45:1–45:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPICS.ICALP.2019.45.

13 Devdatt P Dubhashi and Alessandro Panconesi. *Concentration of measure for the analysis of randomized algorithms*. Cambridge University Press, 2009.

14 Joan Feigenbaum, Sampath Kannan, Andrew McGregor, Siddharth Suri, and Jian Zhang. On graph problems in a semi-streaming model. *Theoretical Computer Science*, 348(2-3):207–216, 2005. doi:10.1016/J.TCS.2005.09.013.

15 Bjarni V Halldórsson, Magnús M Halldórsson, Elena Losievskaja, and Mario Szegedy. Streaming algorithms for independent sets. In *Automata, Languages and Programming: 37th International Colloquium, ICALP 2010, Bordeaux, France, July 6-10, 2010, Proceedings, Part I 37*, pages 641–652. Springer, 2010. doi:10.1007/978-3-642-14165-2_54.

16 Bjarni V Halldórsson, Magnús M Halldórsson, Elena Losievskaja, and Mario Szegedy. Streaming algorithms for independent sets in sparse hypergraphs. *Algorithmica*, 76:490–501, 2016. doi:10.1007/S00453-015-0051-5.

17 Magnús M Halldórsson, Xiaoming Sun, Mario Szegedy, and Chengu Wang. Streaming and communication complexity of clique approximation. In *International Colloquium on Automata, Languages, and Programming*, pages 449–460. Springer, 2012. doi:10.1007/978-3-642-31594-7_38.

18 Richard M Karp. *Reducibility among combinatorial problems*. Springer, 2010. doi:10.1007/978-3-540-68279-0_8.

19 Ami Paz and Gregory Schwartzman. A $(2+\varepsilon)$-approximation for maximum weight matching in the semi-streaming model. *ACM Transactions on Algorithms (TALG)*, 15(2):1–15, 2018. doi:10.1145/3274668.

20 David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 681–690, 2006. doi:10.1145/1132516.1132612.