Constructing Long Paths in Graph Streams

Christian Konrad
School of Computer Science, University of Bristol, UK

Chhaya Trehan
Unaffiliated Researcher, Durham, UK

Abstract

In the *graph stream model of computation*, an algorithm processes the edges of an n-vertex input graph in one or more sequential passes while using a memory that is sublinear in the input size. The streaming model poses significant challenges for algorithmically constructing long paths. Many known algorithms that are tasked with extending an existing path as a subroutine require an entire pass over the input to add a single additional edge. This raises a fundamental question: Are multiple passes inherently necessary to construct paths of non-trivial lengths, or can a single pass suffice? To address this question, we systematically study the Longest Path problem in the one-pass streaming model. In this problem, given a desired approximation factor α, the objective is to compute a path of length at least $\mathrm{lp}(G)/\alpha$, where $\mathrm{lp}(G)$ is the length of a longest path in the input graph G.

We study the problem in the insertion-only and the insertion-deletion streaming models, and we give algorithms as well as space lower bounds for both undirected and directed graphs. Our results are:

1. We show that for undirected graphs, in both the insertion-only and the insertion-deletion models, there are *semi-streaming algorithms*, i.e., algorithms that use space $O(n \operatorname{poly} \log n)$, that compute a path of length at least $d/3$ with high probability, where d is the average degree of the input graph. These algorithms can also yield an α-approximation to Longest Path using space $\tilde{O}(n^2/\alpha)$.
2. Next, we show that such a result cannot be achieved for directed graphs, even in the insertion-only model. We show that computing a $(n^{1-o(1)})$-approximation to Longest Path in directed graphs in the insertion-only model requires space $\Omega(n^2)$. This result is in line with recent results that demonstrate that processing directed graphs is often significantly harder than undirected graphs in the streaming model.
3. We further complement our results with two additional lower bounds. First, we show that semi-streaming space is insufficient for small constant factor approximations to Longest Path for undirected graphs in the insertion-only model. Last, in undirected graphs in the insertion-deletion model, we show that computing an α-approximation requires space $\Omega(n^2/\alpha^3)$.

2012 ACM Subject Classification Theory of computation → Streaming models; Mathematics of computing → Paths and connectivity problems

Keywords and phrases Longest Path Problem, Streaming Algorithms, One-way Two-party Communication Complexity

Digital Object Identifier 10.4230/LIPIcs.ESA.2025.22

Related Version *Full Version*: https://arxiv.org/abs/2508.16022

Funding *Christian Konrad*: Supported by EPSRC New Investigator Award EP/V010611/1.
Chhaya Trehan: Most of the work was done while C.T. was at the University of Bristol where she was supported by EPSRC New Investigator Award EP/V010611/1.

1 Introduction

In the *graph stream model of computation*, an algorithm processes a stream of edge insertions and deletions that make up an n-vertex input graph $G = (V, E)$ via one or multiple sequential passes. The primary objective is to design algorithms that use as little space as possible. The most studied and best-understood setting is the one-pass insertion-only setting, where

only a single pass is allowed and the input stream does not contain any edge deletions. It is known that many fundamental problems can be solved well in this setting, e.g., there are *semi-streaming algorithms*, i.e., algorithms that use space $\tilde{O}(n) = O(n \operatorname{poly} \log n)$[1], for computing a spanning tree and a maximal matching [16], while no $o(n^2)$ space algorithms exist for other problems such as computing a maximal independent set or a BFS/DFS tree [14, 6]. For such problems, algorithms that make multiple passes over the input data are typically considered [24, 13, 1, 9, 4].

Our work is motivated by the observation that streaming algorithms that construct long paths, often as a subroutine, require a large number of passes. For example, many streaming algorithms for computing large matchings construct augmenting paths as a subroutine (e.g., [29, 28, 30]), and these algorithms typically only add a single edge or a few edges per pass to a not-yet-completed augmenting path. Another example is streaming algorithms for computing BFS/DFS trees that extend partial trees/paths by adding only a single edge per pass [24, 13]. This raises a fundamental question: Is the one/few-edges-per-pass strategy best possible and multiple passes are inherently necessary to construct paths of non-trivial lengths, or can a single pass suffice?

The Longest Path Problem. We address this question by systematically studying the Longest Path (LP) problem in the one-pass streaming setting. In this problem, the objective is to compute a path of length $\operatorname{lp}(G)/\alpha$, where $\operatorname{lp}(G)$ denotes the length of a longest path in the input graph G, and $\alpha \geq 1$ is the approximation factor.

In the offline setting, in undirected graphs, it is known that it is NP-hard to compute a constant factor approximation to Longest Path [23] (see the same paper for stronger impossibility results that are based on other hardness conjectures). In directed graphs, a much stronger hardness result is known. Björklund, Husfeldt and Khanna [11] showed that it is NP-hard to approximate LP in directed graphs within a factor of $n^{1-\epsilon}$, for any $\epsilon > 0$. Regarding upper bounds, it is known that a path of length $d_{\min}$ can be constructed greedily in polynomial time, where $d_{\min}$ is the min-degree of the input graph [23]. There are also FPT algorithms with runtime polynomial in n but exponential in the length of the path constructed (see, for example, [10] and the references therein). We note that streaming algorithms for the LP problem have also previously received attention, however, only from a practical perspective. Kliemann et al. [25] studied practical multi-pass algorithms without providing any theoretical guarantees.

1.1 Our Results

In this work, we give one-pass streaming algorithms and space lower bounds for undirected graphs as well as a strong space lower bound for directed graphs. We consider both the insertion-only model (no deletions) and the insertion-deletion model (deletions allowed).

As our first result, we show that, in undirected graphs, if we sample $O(n \log n)$ random edges from the input graph, then this sample contains a path of length $\Omega(d)$ with high probability, where d is the average degree of the input graph. Since this sampling task can be implemented in both the insertion-only and the insertion-deletion streaming models using standard techniques, we obtain the following theorem:

[1] We write $\tilde{O}(.)$ to mean the usual Big-O notation with poly-logarithmic dependencies suppressed. $\tilde{\Theta}(.)$ and $\tilde{\Omega}(.)$ are defined similarly.

▶ **Theorem 1.** *In both the insertion-only and the insertion-deletion models, for undirected graphs, there are $O(n \operatorname{poly} \log n)$ space algorithms that compute a path of length at least $d/3$ with high probability, where d is the average degree of the input graph.*

We remark that our algorithm can be used to obtain an α-approximation to LP using $\tilde{O}(n^2/\alpha)$ space, for any $\alpha \geq 1$. This is achieved by running our algorithm in parallel with the trivial algorithm that stores $\tilde{\Theta}(n^2/\alpha)$ edges. Then, if the space constraint of $\tilde{\Theta}(n^2/\alpha)$ is large enough to store the entire graph then we can find an exact solution in exponential time. If not, then we are guaranteed that the average degree of the input graph is $\Omega(n/\alpha)$, which implies that the algorithm of Theorem 1 yields the desired result. We thus obtain the following corollary:

▶ **Corollary 2.** *In both insertion-only and insertion-deletion models, for undirected graphs, there are $\tilde{O}(n^2/\alpha)$-space streaming algorithms for an α-approximation to Longest Path.*

Next, we ask whether a similar algorithmic result is possible in directed graphs. As our main and most technical result, we show that this is not the case in a strong sense:

▶ **Theorem 3.** *Every one-pass streaming algorithm for Longest Path in the insertion-only model in directed graphs with approximation factor $n^{1-o(1)}$ requires space $\Omega(n^2)$.*

Theorem 3 together with Corollary 2 establishes a separation in the space complexity between algorithms for undirected and directed graphs for LP in the insertion-only model. This lower bound is also in line with recent results for directed graphs that demonstrate that problems on general directed graphs are often hard to solve in the streaming model [12].

Finally, we complement our results with two additional lower bounds. First, we give a lower bound for insertion-only streams and undirected graphs, ruling out the existence of semi-streaming algorithms with constant approximation factor close to 1.

▶ **Theorem 4.** *Every one-pass insertion-only streaming algorithm for Longest Path on undirected graphs with approximation factor $1 + \frac{1}{25} - \gamma$, for any $\gamma > 0$, requires space $n^{1+\Omega(\frac{1}{\log \log n})}$.*

We note that this lower bound result is significantly weaker than our lower bound for directed graphs, both in terms of approximation factor and space. This, however, is in line with the status of the problem in the offline setting, where a very strong impossibility result for directed graphs is known, but only significantly weaker impossibility results for undirected graphs exist. While the offline setting is orthogonal to the streaming setting, one cannot help but wonder whether similar mechanisms are at work that prevent us from obtaining stronger NP-hardness results for LP approximation in undirected graphs and from obtaining stronger space lower bounds in the streaming setting.

Last, we show that, in insertion-deletion streams, space $\Omega(n^2/\alpha^3)$ is required for an α-approximation of LP.

▶ **Theorem 5.** *Every one-pass insertion-deletion streaming algorithm for Longest Path on undirected graphs with approximation factor $\alpha \geq 1$ requires space $\Omega(n^2/\alpha^3)$.*

This lower bound together with our algorithm show that the optimal dependency of the space complexity on α in insertion-deletion streams is between $1/\alpha$ and $1/\alpha^3$.

1.2 Our Techniques

Algorithm. We will first explain the key ideas behind our algorithm. As observed by Karger et al. [23], a path of length $d_{\min}$, where $d_{\min}$ is the minimum degree of the input graph G, can be computed as follows: Start at any vertex v_0 and visit an arbitrary neighbor that we denote by v_1. In a general step i, we have already constructed the path $v_0, \ldots, v_i$. We then visit a neighbor v_{i+1} of v_i that has not previously been visited. Then, as long as $i < d_{\min}$, we can always find a yet unvisited neighbor, and, hence, we obtain a path of length $d_{\min}$. We first see that this argument can also be applied to the average degree d of the graph G. It is well known that, by repeatedly removing vertices of degree at most $d/2$ from G until no such vertex remains, we are left with a non-empty graph G' with min-degree at least $d/2$. We can now apply the same argument as above to G' and thus find a path of length at least $d/2$.

The approach outlined above, however, does not yield a small space streaming algorithm since a) we do not know which vertices are contained in G', and b) we cannot afford to store $\Theta(d)$ incident edges on each vertex. To overcome these obstacles, we resort to randomization. Our algorithm solely samples $O(n \log n)$ random edges F from the input graph and outputs a longest path among the edges F. To see that a long path in F exists, we argue that the subset of edges $F' \subseteq F$ that are also contained in G' contains a path of length at least $d/3$ with high probability. Such a path can be constructed greedily. Suppose we have already constructed a partial path of length $\ell < d/3$ solely using the edges F', and let $v_{\ell+1} \in V(G')$ denote its current endpoint. Then, since $v_{\ell+1}$ has a degree of at least $d/2$ in G', there are at least $d/2 - d/3 = d/6$ neighbors of $v_{\ell+1}$ in G' that have not yet been visited in the path. Since we sample $\Theta(n \log n)$ edges overall, the probability that any one of these edges incident to $v_{\ell+1}$ that connect to these $d/6$ vertices is sampled is $\Omega(\frac{n \cdot \log n}{n \cdot d}) = \Omega(\frac{\log n}{d})$, which implies that at least one of these edges is sampled with high probability and we can extend the path.

Space Lower Bounds. All our space lower bounds are proved in the one-way two-party communication setting. In this setting, two parties that we denote by Alice and Bob each hold a subset of the edges of the input graph $G = (V, E = E_A \cup E_B)$, with E_A being Alice's edges, and E_B being Bob's edges. Alice sends a single messages Π to Bob, and Bob computes the output of the protocol. Then, it is well-known that a lower bound on the size of the message Π also constitutes a lower bound on the space of any one-pass streaming algorithm.

We work with *induced matchings* in all our lower bound constructions. In a graph $G = (V, E)$, a matching $M \subseteq E$ is *induced* if the edges of the vertex-induced subgraph $G[V(M)]$ are precisely the edges of M. Suppose now that G is bipartite, and we denote it by $G = (A, B, E = E_A \cup E_B)$. Furthermore, we suppose that Alice's subgraph, i.e., the graph spanned by the edges E_A, contains a matching M that is induced in the final graph. We say that M is the *special* matching. We complete our lower bound constructions so that:

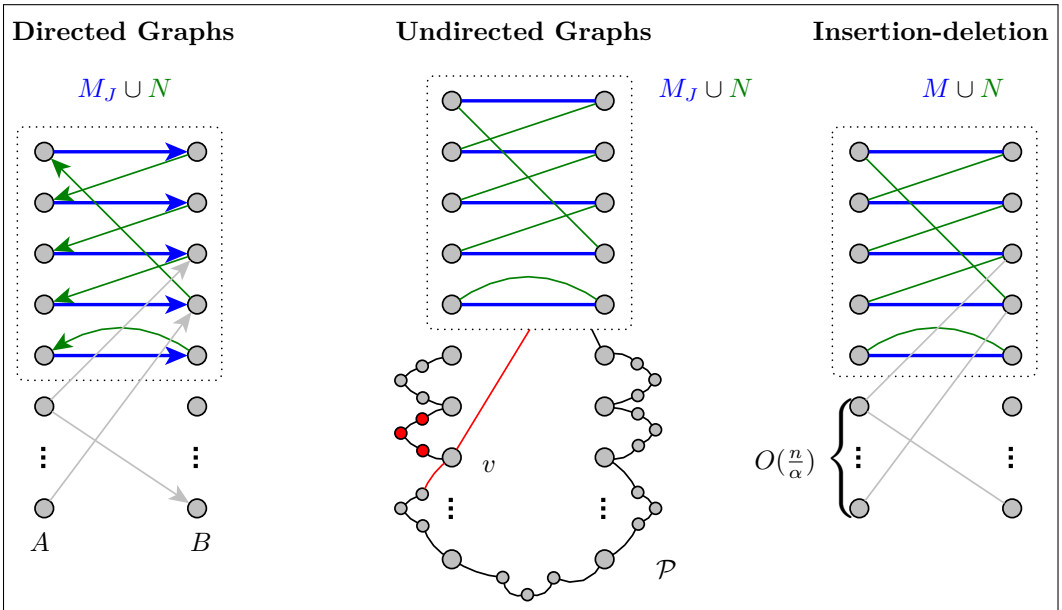
1. Every long path in the graph must contain many edges of the special matching M;
2. The special matching M is *hidden* among the edge set E_A so that Alice cannot identify M, and, given a limited communication budget, Alice therefore cannot forward many of M's edges to Bob.

The two properties then imply a lower bound since, if Bob knows only few edges of M, but every long path contains many such edges, then Bob cannot output a long path.

To achieve property 2, in our insertion-deletion lower bound, we make use of edge deletions as part of Bob's input to turn a large matching in E_A into an induced matching, and in our insertion-only lower bounds, we work with *Rusza-Szemerédi graphs* (RS-graphs in short), which have been extensively used for proving lower bounds for matching problems in the streaming setting (e.g., [18, 8, 26, 27]). An (r, t)-RS-graph is a balanced bipartite graph on $2n$

vertices such that its edge set can be partitioned into t induced matchings $M_1, \ldots, M_t$, each of size r. Our insertion-only constructions are such that each of these t matchings can take the role of the special matching M, which will allow us to argue that Alice essentially has to send many edges of each induced matching to Bob if Bob is to report a long path. We note that the framework of working with RS-graphs and a special hidden matching among Alice's edges is well-established and has previously been used, for example, in [18, 7, 3, 27, 5, 4].

Regarding establishing property 1, we pursue different strategies that depend on the specific streaming setting, and on whether we work with directed or undirected graphs. These strategies are described below and illustrated in Figure 1.



Directed Graphs: Since N constitutes the only edges directed from right to left, and M_J is an induced matching, up to potentially the first and last edge, every long path alternates between edges of M_J and N and thus contains many edges of M_J.

Undirected Graphs: (Insertion-Only) The path $\mathcal{P}$ is connected to a longest path in $V(M_J)$. Our argument is based on the fact that it is detrimental to the length of the output path to include many edges between $V \setminus V(M_J)$ and $V(M_J)$. Indeed, suppose that the output path contains the two red edges incident on vertex v. Then, the vertices highlighted in red cannot be visited by the output path. This allows us to argue that, for each edge going across the cut $(V \setminus V(M_J), V(M_J))$, multiple vertices on the path $\mathcal{P}$ are not visited. This in turn implies that every long path must contain many edges of M_J in order to visit the vertices in $V(M_J)$.

Insertion-Deletion Streams: (Undirected Graphs) Our construction is such that there are only $O(\frac{n}{\alpha})$ vertices outside $V(M)$. Hence, if Bob does not know many edges of M then there are at most $O(\frac{n}{\alpha})$ vertices of $V \setminus V(M)$ available to connect to vertices within $V(M)$. Since this number is small, Bob is required to learn many edges of M in order to output a long path.

Figure 1 Illustrations of our three lower bound constructions.

Directed Graphs in the Insertion-only Model. The cleanest case is our lower bound for directed graphs. In this setting, Alice holds the edges of an RS-graph $G' = (A, B, E)$, and we assume that all edges are directed from A to B. We then pick a matching M_J, for a uniform random $J \in [t]$, and Bob inserts a random matching N that matches $B(M_J)$ to $A(M_J)$ so that all edges in N are directed from B to A. We then leverage the well-known result that the expected length of a longest cycle in a random permutation of the set $[r]$ is of length $\Theta(r)$ [20, 19] in order to argue that $M_J \cup N$ contains a path of expected length $\Theta(|M_J|)$. It then remains to argue that every path of length ℓ, for any ℓ, in the input graph $G = G' \cup N$ must contain $\Omega(\ell)$ edges of M_J, which uses the fact that the matching M_J is induced. Since, however, Alice did not know that M_J is the special matching, Alice is required to send a large number of edges of each induced matching to Bob so that Bob can output a long path.

For technical reasons, our graph construction is slightly more involved than described above since we need to turn Alice's RS-graph into an input distribution, see Section 4 for details. While it is easy to argue that if Alice sends k edges of M_J to Bob then Bob can compute a path of length at most $O(k)$ in $M_J \cup N$, we need to argue a much stronger bound. We show that, even if Alice sends as many as $r/100$ M_J edges to Bob then Bob can still only find a path of size $O(\log r)$ in $M_J \cup N$. We implement our approach using the information complexity paradigm, including direct sum and message compression arguments.

Undirected Graphs in the Insertion-only Model. In our construction for undirected graphs, Alice holds an RS-graph $G' = (A, B, E)$, and, given a random index $J \in [t]$, Bob holds a random matching N matching $A(M_J)$ to $B(M_J)$. Similar as above, the matchings $M_J \cup N$ contain a path of length $\Omega(|M_J|)$. However, it is no longer true that any long path in $G' \cup N$ must contain many edges of M_J as, for example, the vertices in $V \setminus V(M_J)$ can now be used to visit the vertices within $V(M_J)$ as their incident edges are undirected.

Our aim is to ensure that it is detrimental for constructing long paths to include many edges across the cut $(V \setminus V(M_J), V(M_J))$ in the path. Once this property is established, we will argue that, if Bob only knows few of the M_J edges then many vertices of $V(M_J)$ will remain unvisited in the output path, which implies that the path is bounded in length as not all vertices are visited. To argue that only few edges across the cut $(V \setminus V(M_J), V(M_J))$ are included in the output path, we ensure that all the vertices $v \in V \setminus V(M_J)$ serve as *gateways* to other parts of the graph that are introduced by Bob. The construction is so that these other parts cannot be visited if v connects to a vertex in $V(M_J)$ in the output path, which will then be detrimental for the construction of a long path. To achieve this, Bob introduces additional edges as follows. First, let $\mathcal{P}'$ be a path consisting of novel edges that visits every vertex in $V \setminus V(M_J)$, and let $\mathcal{P}$ be the path obtained from $\mathcal{P}'$ by subdividing every edge in $\mathcal{P}'$, ℓ times, for some integer ℓ. Furthermore, this path is connected to the longest path within $M_J \cup N$. Then, the path $\mathcal{P}$ significantly contributes to the longest path in the input graph. Observe, however, if a vertex $v \in V \setminus V(M_J)$ connects to a vertex in $V(M_J)$ then the vertices on the subdivision on one of the edges of $\mathcal{P}$ incident on v cannot be visited anymore. By setting ℓ large enough, we show that it is detrimental for the algorithm to use such vertices to connect to $V(M_J)$, which renders the edges of M_J indispensable for a long path. As above, the actual lower bound construction is more involved since we need to turn Alice's RS-graph into an input distribution. On a technical level, we give a reduction to the two-party communication problem Index using ideas similar to those introduced by Dark and Konrad [15].

Undirected Graphs in the Insertion-deletion Model. The key advantage that allows us to prove a much stronger lower bound for insertion-deletion streams than for insertion-only streams is that Bob can insert edge deletions that turn a large matching contained in Alice's edges E_A into an induced matching. We see that it is possible to achieve this so that:

1. The special (induced) matching M is of size $n - O(n/\alpha)$; and
2. The special matching M is hidden among Alice's edges.

Furthermore, we make sure that there exists a second matching N such that $N \cup M$ forms a path of length $\Omega(n)$. Property 1 significantly helps in proving our lower bound since, as opposed to our lower bound for undirected graphs in insertion-only streams, there are only $O(n/\alpha)$ vertices outside the set $V(M)$, and, hence, these vertices only allow us to visit $O(n/\alpha)$ vertices of $V(M)$. We can then argue that, for the remaining $n - O(n/\alpha)$ vertices of $V(M)$, Bob is required to know the edges of M for those to be visited. This however requires Alice to send these edges to Bob. Then, Property 2 ensures that Alice cannot identify these edges and would therefore have to send most edges of the input graph to Bob, which exceeds the allowed communication budget. On a technical level, we give a reduction to the Augmented-Index two-party communication problem, again, reusing many of the ideas given in the lower bound by Dark and Konrad [15].

1.3 Outline

We provide notation, state RS-graph constructions, introduce the information complexity framework, state important inequalities involving mutual information, and give a message compression theorem in our preliminaries section, Section 2. Then, we present our algorithm in Section 3, and our lower bound for directed graphs in insertion-only streams is given in Section 4. Due to space restrictions, our lower bound for undirected graphs in insertion-only streams and our lower bound for undirected graphs in insertion-deletion streams are deferred to the full version of this paper. We conclude in Section 5 with open problems.

2 Preliminaries

Given a graph $G = (V, E)$, for vertices $u, v \in V$, we denote an undirected edge between u and v by $\{u, v\}$, and a directed edge with tail u and head v by (u, v).

2.1 Rusza-Szemerédi Graphs

In our lower bounds, we make use of Rusza-Szemerédi graphs.

▶ **Definition 6** (Rusza-Szemerédi Graph). *A bipartite graph $G = (A, B, E)$ is an (r, t)-*Ruzsa-Szemerédi graph*, (r, t)-RS graph for short, if its edge set can be partitioned into t induced matchings, each of size r.*

We will use the RS-graph constructions of Alon et al. [2] and Goel et al. [18], the latter is based on the construction by Fischer et al. [17].

▶ **Theorem 7** (Ruzsa-Szemerédi Graph Constructions). *There are bipartite RS-graphs $G = (A, B, E)$ with $|A| = |B| = n$ with the following parameters:*

1. *$r = n^{1-o(1)}$, and $t \cdot r = \Omega(n^2)$ [2]; and*
2. *$r = (\frac{1}{2} - \epsilon)n$, for any constant $\epsilon > 0$, and $t = n^{\Omega(\frac{1}{\log \log n})}$ [18].*

2.2 Streaming Models

Given an input graph $G = (V, E)$, an *insertion-only* stream describing G is an arbitrarily ordered sequence of the edge set E. An *insertion-deletion* stream is a sequence of edge insertions and deletions so that, at the end of the stream, the surviving edges constitute the edge set E. Furthermore, the stream is such that only edges that have previously been

inserted are deleted, i.e., the multiplicity of an edge is never negative. We also assume that at any moment the multiplicity of any edge is polynomially bounded. This standard assumption ensures that sampling methods such as ℓ_0-sampling require only $\operatorname{poly}\log(n)$ space.

2.3 Communication Complexity

We consider the one-way two-party model of communication for proving our space lower bounds. In this setting, two parties, denoted Alice and Bob, share the input data $X = (X_A, X_B)$ so that Alice holds X_A and Bob holds X_B. They operate as specified in a protocol Π in order to solve a problem $\mathcal{P}$. Alice and Bob can make use of both private and public randomness. Randomness is provided via infinite sequences of uniform random bits. Private randomness can only be accessed by one party. The sequence of public randomness can be accessed by both Alice and Bob, and we denote this sequence by R.

The protocol Π instructs the parties to operate as follows. First, Alice computes a message that we (ambiguously) also denote by Π as a function of X_A, R, and her private randomness, sends the message Π to Bob, who then computes the output of the protocol as a function of X_B, Π, R and his private randomness. The *communication cost* of a protocol Π is the maximum length of the message sent by Alice in any execution of Π. The *communication complexity* of a problem $\mathcal{P}$ is the minimum communication cost of any protocol that solves $\mathcal{P}$.

2.4 Information Complexity and Message Compression

We prove lower bounds using the *information complexity paradigm*, which is a framework that is based on information theory. Let $(A, B, C) \sim \mathcal{D}$ be jointly distributed random variables according to distribution $\mathcal{D}$. We denote the *Shannon entropy* of A by $H_{\mathcal{D}}(A)$, the entropy of A conditioned on B by $H_{\mathcal{D}}(A \mid B)$, the *mutual information* of A and B by $I_{\mathcal{D}}(A \ : \ B)$, and the conditional mutual information between A and B conditioned on C by $I_{\mathcal{D}}(A \ : \ B \mid C)$. We may drop the subscript $\mathcal{D}$ in $H_{\mathcal{D}}(.)$ and $I_{\mathcal{D}}(.)$ if it is clear from the context.

We will use the following standard facts about entropy and mutual information: (let $(A, B, C, D) \sim \mathcal{D}$ be jointly distributed random variables.)

P1: If A and C are independent conditioned on D then: $I(A \ : \ B \mid D) \leq I(A \ : \ B \mid C, D)$
P2: $I(A \ : \ B \mid C, D) = \mathbb{E}_{d \leftarrow D} I(A \ : \ B \mid C, D = d)$
P3: Let E be an event independent of A, B, C. Then: $I(A \ : B \mid C, E) = I(A \ : B \mid C)$
P4: $I(A, B \ : C \mid D) = I(A \ : C \mid D) + I(B \ : C \mid D, A)$

Given a one-way two-party communication protocol Π and an input distribution $(X_A, X_B) \sim \mathcal{D}$, we will measure the amount of information that the message Π reveals about Alice's input under distribution $\mathcal{D}$. The following quantity is denoted the *external information cost* of Π:

▶ **Definition 8.** *The (external) information cost* $\mathrm{ICost}_{\mathcal{D}}(\Pi)$ *of the one-way two-party communication protocol* Π *under input distribution* $\mathcal{D}$ *is defined as:*

$$\mathrm{ICost}_{\mathcal{D}}(\Pi) = I_{\mathcal{D}}(X_A \ : \ \Pi \mid R) \ .$$

Then, for a given problem $\mathcal{P}$, we denote the *information complexity* $\mathrm{IC}_{\mathcal{D}}(\mathcal{P})$ of $\mathcal{P}$ under distribution $\mathcal{D}$ as the minimum information cost of a protocol that solves $\mathcal{P}$.

It is well-known that information cost of a protocol lower bounds its communication cost.

We will also use a *message compression* result, which is due to Harsha et al. [21]. We follow the presentation of this result given in [31].

▶ **Theorem 9** (Message Compression). *Let Π be a protocol in the one-way two-party communication setting. Then, the protocol can be simulated with a different protocol that sends a message of expected size at most*

$$I_{\mathcal{D}}(X_A \ : \ \Pi) + 2 \cdot \log\left(1 + I_{\mathcal{D}}(X_A \ : \ \Pi)\right) + O(1) \ .$$

3 Algorithm

We will first describe and analyze our sampling-based algorithm in Subsection 3.1, and then discuss implementations of this algorithm in streaming models in Subsection 3.2.

3.1 Sampling-based Algorithm

Let $G = (V, E)$ be the input graph with $n = |V|$, $m = |E|$, and average degree $d = 2\frac{m}{n}$. Consider the following algorithm:

Algorithm 1 Sampling-based algorithm for constructing a path of length $\Omega(d)$.

Require: $G = (V, E)$
Sample $10 \cdot n \cdot \ln n$ random edges from E and denote this set by F
return Longest path in $G[F]$

We show that Algorithm 1 constructs a path of length at least $d/3$ with high probability.

▶ **Theorem 10.** *Algorithm 1 constructs a path of length $d/3$ with probability at least $1 - \frac{1}{n^2}$. It can also be regarded as an $O(\frac{n}{d})$-approximation algorithm for Longest Path.*

Proof. Given the input graph $G = (V, E)$, let $U \subseteq V$ denote a subset of vertices of V such that $G[U]$ has minimum degree at least $d/2$. It is well-known that such a subset of vertices exists (see Lemma 19 for completeness).

Let $u_0 \in U$ be any vertex, and let $P_0 = \{u_0\}$ be the path of length 0 with start and end point u_0. We will extend P_0 using the edges in F that are also contained in $G[U]$, as follows:

In step $i = 1, 2, \ldots, d/3$, we add the edge (u_{i-1}, u_i) to P_{i-1} and obtain the path P_i. The edge (u_{i-1}, u_i) is an arbitrary edge in F incident on u_{i-1} that connects to a vertex $u_i \in U$ that has not yet been visited on the path P_{i-1}. We will now prove that such a vertex exists.

First, recall that, by definition of U, the vertex u_{i-1} has at least $d/2$ neighbors in $G[U]$. Since $i \leq d/3$, at least $d/2 - d/3 = d/6$ of these neighbors are not visited by the path P_{i-1}. Denote by $E(u_{i-1})$ this set of at least $d/6$ edges connecting u_{i-1} to not yet visited neighbors in $G[U]$. We now claim that at least one of the edges of $E(u_{i-1})$ is contained in F with high probability. Observe that at stage $i-1$, we have only learnt so far that the sample F contains the $i-1$ edges of the path P_{i-1}. Hence,

$$\begin{aligned}
&\Pr\left[F \cap E(u_{i-1}) = \varnothing \mid P_{i-1} \subseteq F\right] = \frac{\binom{m-(i-1)-|E(u_{i-1})|}{|F|-(i-1)}}{\binom{m-(i-1)}{|F|-(i-1)}} \\
&\leq \exp\left(-\frac{|E(u_{i-1})|(|F| - (i-1))}{m-(i-1)}\right) \leq \exp\left(-\frac{\frac{d}{6}(10n\ln(n) - n\ln(n))}{m}\right) \\
&= \exp(-\frac{\frac{m}{3n}(9n\log n)}{m}) = \frac{1}{n^3} \ ,
\end{aligned}$$

where we used the inequality $\frac{\binom{a-c}{b}}{\binom{a}{b}} \leq \exp\left(-\frac{bc}{a}\right)$ (see Lemma 18) to obtain the first inequality and the bound $(i-1) \leq n\ln(n)$ to obtain the second.

We can therefore extend the path with probability $1 - \frac{1}{n^3}$ at any step i. Since we run $d/3 \leq n$ steps overall to create the final path $P_{d/3}$ of length $d/3$, by the union bound, we succeed with probability at least $1 - \frac{1}{n^2}$. Last, since a longest path in G is of length at most n, the algorithm also constitutes an $O(n/d)$-approximation algorithm to Longest Path. ◀

3.2 Implementation in Streaming Models

Algorithm 1 can easily be implemented in both the insertion-only and the insertion-deletion streaming models. In the insertion-only model, a uniform sample of the edges in the stream can be obtained using *reservoir sampling* [32], and, in the insertion-deletion model, this can be achieved using ℓ_0-samplers [22] and rejection sampling. We obtain our main theorem:

▶ **Theorem 1.** *In both the insertion-only and the insertion-deletion models, for undirected graphs, there are semi-streaming algorithms that compute a path of length at least $d/3$ with high probability, where d is the average degree of the input graph.*

4 Insertion-only Lower Bound for Directed Graphs

In this section, we prove an $\Omega(n^2)$ space lower bound for one-pass streaming algorithms for LP on directed graphs that compute an $(n^{1-o(1)})$-approximation. To this end, we work with two input distributions $\mathcal{D}_{\text{SLP}}$ (**S**imple **L**ongest **P**ath) and $\mathcal{D}_{\text{LP}}$ (**L**ongest **P**ath). In Subsection 4.1, we give a lower bound on the information cost of protocols that solve $\mathcal{D}_{\text{SLP}}$ well. We achieve this by, first, proving a lower bound on the communication cost of any protocol that solves $\mathcal{D}_{\text{SLP}}$ directly via combinatorial arguments, and then employ a message compression argument that allows us to conclude that the information cost of such protocols must also be large. Then, in Subsection 4.2, we present the distribution $\mathcal{D}_{\text{LP}}$, which makes use of an (r, t)-RS-graph, and we establish a direct sum argument, showing that the information cost of protocols that solve $\mathcal{D}_{\text{LP}}$ is at least t times the information cost of protocols that solve $\mathcal{D}_{\text{SLP}}$, which bounds the information cost of protocols that solve $\mathcal{D}_{\text{LP}}$ from below. Then, since information cost is a lower bound on communication cost, we obtain our result.

4.1 A Simple Distribution

We will first work with the distribution denoted $\mathcal{D}_{\text{SLP}}(r)$, see Figure 2.

We first argue that the expected length of a longest path in $H \sim \mathcal{D}_{\text{SLP}}(r)$ is $\Omega(r)$.

▶ **Lemma 11.** *The expected length of a longest path in $H \sim \mathcal{D}_{SLP}(r)$ is bounded as follows:*

$$\mathop{\mathbb{E}}_{H \leftarrow \mathcal{D}_{SLP}(r)} \text{lp}(H) \geq 2\lambda r \geq 1.24r \ ,$$

where $\lambda = 0.62432\ldots$ is the Golomb-Dickman constant.

Proof. For an input graph $H = (A, B_1 \cup B_2, E) \sim \mathcal{D}_{\text{SLP}}(r)$ with $B_1 = \{b_1^1, \ldots, b_r^1\}$ and $B_2 = \{b_1^2, \ldots, b_r^2\}$, let H' be the graph obtained from H by *contracting* the vertex pairs b_i^1 and b_i^2, for all i, and by treating parallel edges in the resulting graph as single edges, see Figure 3 for an illustration. It is easy to see that $\text{lp}(H') = \text{lp}(H)$. Furthermore, denote by $\mathcal{D}'_{\text{SLP}}(r)$ the distribution of H'. Observe that $\mathcal{D}_{\text{SLP}}(r)$ and $\mathcal{D}'_{\text{SLP}}(r)$ are uniform distributions over their respective supports.

Next, consider the set Σ_r of permutations of the set $\{1, 2, \ldots, r\}$. Consider now the bijection $f : \Sigma_r \to \text{range}(\mathcal{D}'_{\text{SLP}}(r))$, where a permutation $\sigma \in \Sigma_r$ is mapped to the graph that contains the edges $(b_i, a_{\sigma(i)})$, for all i. Then, we observe that, for a permutation $\sigma \in \Sigma_r$, the length of the longest cycle $\text{lc}(\sigma)$ is related to $\text{lp}(f(\sigma))$ as follows:

$$2 \cdot \text{lc}(\sigma) - 1 = \text{lp}(f(\sigma)) \ .$$

Input Distribution $\mathcal{D}_{\text{SLP}}(r)$:

The directed graph $G = (A, B = B_1 \cup B_2, E)$ with $|A| = |B_1| = |B_2| = r$, $A = \{a_1, \ldots, a_r\}$, $B_1 = \{b_1^1, \ldots, b_r^1\}$, $B_2 = \{b_1^2, \ldots, b_r^2\}$, and $E = E_A \cup E_B$, where E_A are Alice's edges and E_B are Bob's edges, is obtained as follows:

Alice's Input: Edge set E_A

For each $i \in [r]$, flip an unbiased coin $X_i \in \{0, 1\}$ and if it comes out heads then insert the directed edge (a_i, b_i^1) into the graph. If it comes out tail then insert the edge (a_i, b_i^2). Observe that this constitutes a directed matching M that matches all A-vertices and exactly one of b_i^1, b_i^2, for all i.

Alice holds the edges $E_A = M$.

Bob's Input: Edge set E_B

Let N^1 be a uniform random matching between A and B_1, directed from B_1 towards A. Let N^2 be a copy of N^1, but every B_1-vertex is replaced by the corresponding B_2-vertex.

Bob holds the edges $E_B = N^1 \cup N^2$.

Figure 2 Input Distribution $\mathcal{D}_{\text{SLP}}(r)$.

It is known that the expected length of a longest cycle in a random permutation $\sigma \in \Sigma_r$ is at least $\lambda \cdot r$, where $\lambda = 0.624\ldots$ is the Golomb-Dickman constant [20, 19]. Hence, we obtain

$$\mathop{\mathbb{E}}_{\mathcal{H}' \sim \mathcal{D}'_{\text{SLP}}(r)} \text{lp}(H') \geq 2 \cdot \lambda \cdot r - 1 \geq 1.24 \cdot r \ ,$$

using the assumption that r is large enough. Since the longest paths in H and H' are identical, the result follows. ◀

Next, we show that any deterministic protocol on distribution $\mathcal{D}_{\text{SLP}}(r)$ that communicates at most $\frac{1}{100}r$ bits outputs a path of length $O(\log r)$ with high probability over $\mathcal{D}_{\text{SLP}}(r)$.

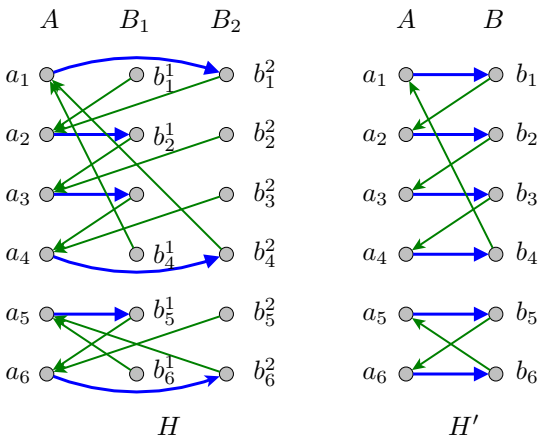
▶ **Lemma 12.** *Let Π_{SLP} be a deterministic one-way communication protocol for Longest Path on distribution $\mathcal{D}_{SLP}(r)$ that communicates at most $\frac{r}{100}$ bits. Then, the probability over the input distribution $\mathcal{D}_{SLP}(r)$ that Π_{SLP} outputs a path of length $O(\log r)$ is at least $1 - \frac{1}{500}$.*

Proof. Denote by $\mathcal{M}$ the set of possible directed input matchings M for Alice in $\mathcal{D}_{\text{SLP}}(r)$. Then, $|\mathcal{M}| = 2^r$. Next, since every message sent by protocol Π_{SLP} is of length at most $s := r/100$, there are at most 2^s different messages. On average, a message therefore corresponds to 2^{r-s} inputs, and, at most $\frac{1}{1000} \cdot 2^r$ inputs yield messages that in turn only correspond to at most $\frac{1}{1000}2^{r-s}$ inputs. Hence, at least $\frac{999}{1000}2^r$ inputs yield messages that each correspond to at least $\frac{1}{1000}2^{r-s}$ inputs. Consider now one of these messages π, and let $M_1, M_2, \ldots$ be the matchings M that correspond to π. Given π, the protocol can only output an edge if it is contained in all matchings M_i. Suppose that there are k such edges. Then, there are at most 2^{r-k} input graphs that contain these edges. We then have:

$$2^{r-k} \geq \frac{1}{1000}2^{r-s} \ ,$$

which implies $k \leq s + 10$.

Denote by K this set of at most $k \leq s + 10$ edges. We will now prove that the longest path in $K \cup E_B$ is of length $O(\log r)$ with high probability.



$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 3 & 4 & 1 & 6 & 5 \end{pmatrix}$$

Figure 3 Illustration of the proof of Lemma 11. The vertices b_i are obtained by contracting b_i^1 and b_i^2. We observe that the permutation σ has a longest cycle of length 4 $(1-2-3-4)$, while H' (and H) have longest paths of lengths 7 $(a_1, b_1, a_2, b_2, \ldots, a_4, b_4)$.

Denote by A' the A-endpoints of the edges K, and let $a'_0 \in A'$ be any vertex. We bound the length of the path with starting point a'_0. This path first uses the edge in K incident on a'_0, and denote by b'_0 the other endpoint of this edge. Then, the path uses the edge of N_1 or N_2 incident on b'_0 to return to an A-vertex that we denote by a'_1. Observe that we can only continue on this path if $a'_1 \in A'$. If this is the case then we can continue in the same fashion, visit a B-vertex b'_1, and return to another A-vertex a'_2. Again, we can only continue if $a'_2 \in A'$, and so on. For any $i \geq 1$, the following holds:

$$\Pr[a'_i \in A' \mid a'_0, \ldots, a'_{i-1} \in A'] \leq \frac{s+10-i}{r-i} \leq \frac{s+10}{r} = \frac{1}{100} + \frac{10}{r} \leq \frac{1}{50} ,$$

assuming that r is large enough. Hence, the probability that the path contains $p+2$ A'-vertices and is thus of length $2(p+1)$ is at most $\frac{1}{50^p}$. This further implies that, with probability at least $1 - \frac{1}{r^2}$, the path is of length $O(\log r)$.

Observe that the argument above applies when considering any start vertex in A'. Hence, by a union bound over all possible start vertices in A', all paths starting at an A' vertex are of length $O(\log r)$ with probability at least $1 - \frac{1}{r}$. Last, observe that a longest path that may be able to form could also start at a B-vertex. Such a path however is only by one edge longer than a path starting at an A-vertex. The $O(\log r)$ length bound therefore still holds.

So far, we proved that for a message that corresponds to at least $\frac{1}{1000}2^{r-s}$ inputs, Bob can only output a path of length at most $O(\log r)$ with high probability, where the probability is over Bob's input. Hence, overall, for a uniform input from $\mathcal{D}_{\mathrm{SLP}}(r)$, the probability that Bob will output a path of length $O(\log r)$ is at least $\frac{999}{1000} \cdot (1 - \frac{1}{r}) \geq \frac{998}{1000}$ for r large enough. ◀

We use the notation $\mathsf{LP}_{\epsilon}^{\alpha}$ to denote LP with approximation factor α and error probability ϵ.

▶ **Theorem 13.** *The randomized one-way communication complexity of* $\mathsf{LP}_{1/4}^{O(r/\log r)}$ *on inputs from* $\mathcal{D}_{SLP}(r)$ *is* $\Omega(r)$.

Proof. Towards a contradiction, let Π_{r} be a randomized protocol for Longest Path on inputs from $\mathcal{D}_{\mathrm{SLP}}(r)$ with approximation factor $o(r/\log r)$ that errs with probability at most $1/4$. Given Π_{r}, by Yao's Lemma, there exists a deterministic protocol Π_{d} on distribution $\mathcal{D}_{\mathrm{SLP}}(r)$ with distributional error $1/4$ and approximation factor $o(r/\log r)$, i.e., on at least $3/4$ of the inputs, the protocol achieves a $o(r/\log r)$-approximation.

Lemma 11 states that $\mathbb{E}_{H \leftarrow \mathcal{D}_{\mathrm{SLP}}(r)} \mathrm{lp}(H) \geq 1.24r$. This allows us to bound the quantity $\Pr_{H \leftarrow \mathcal{D}_{\mathrm{SLP}}(r)}[\mathrm{lp}(H) \geq \frac{1}{2}r]$, as follows:

$$1.24r \leq \mathop{\mathbb{E}}_{H \leftarrow \mathcal{D}_{\mathrm{SLP}}(r)} \mathrm{lp}(H) \leq \Pr_{H \leftarrow \mathcal{D}_{\mathrm{SLP}}(r)}[\mathrm{lp}(H) \geq \frac{1}{2}r] \cdot 2r + (1 - \Pr_{H \leftarrow \mathcal{D}_{\mathrm{SLP}}(r)}[\mathrm{lp}(H) \geq \frac{1}{2}r])\frac{1}{2}r$$
$$= r \cdot \left(1.5 \cdot \Pr_{H \leftarrow \mathcal{D}_{\mathrm{SLP}}(r)}[\mathrm{lp}(H) \geq \frac{1}{2}r] + \frac{1}{2}\right) ,$$

where we used the fact that the longest path in H is at most the number of vertices in H, i.e., $2r$. The previous inequality then implies that $\Pr_{H \leftarrow \mathcal{D}_{\mathrm{SLP}}(r)}[\mathrm{lp}(H) \geq \frac{1}{2}r] \geq \frac{1.24-0.5}{1.5} \geq \frac{1}{3}$.

Next, Lemma 12 states that, with probability at least $1 - \frac{1}{500}$, Π_{d} outputs a path of length at most $O(\log r)$. Hence, with probability at least $1/3 - 1/500 > 1/4$, there simultaneously exists a longest path of length at least $\frac{1}{2}r$ and Π_{d} outputs one of length $(O \log r)$. The approximation factor of Π_{d} is therefore $\Omega(r/\log r)$, contradicting the fact that Π_d achieves an approximation factor of $o(r/\log r)$ on at least $3/4$ of the instances. Protocols Π_d and Π_r therefore do not exist, which completes the proof. ◀

▶ **Lemma 14.** *Let Π be a randomized protocol for $\mathsf{LP}^{O(r/\log r)}_{1/4-\frac{1}{100}}$ on inputs from $\mathcal{D}_{SLP}(r)$. Then:*

$$\mathrm{ICost}_{\mathcal{D}_{SLP}(r)}(\Pi) = \Omega(r) .$$

Proof. Denote by Π a randomized one-way two-party communication protocol for $\mathsf{LP}^{O(r/\log r)}_{1/4-\frac{1}{100}}$ on inputs from $\mathcal{D}_{\mathrm{SLP}}(r)$. Given Π, using the message compression technique stated in Theorem 9, we obtain a protocol Π' for $\mathsf{LP}^{O(r/\log r)}_{1/4-\frac{1}{100}}$ that sends a message of expected size (recall that M is Alice's input matching)

$$s := I_{\mathcal{D}_{\mathrm{SLP}}(r)}(M \ : \ \Pi) + 2 \cdot \log\left(1 + I_{\mathcal{D}_{\mathrm{SLP}}(r)}(M \ : \ \Pi)\right) + O(1) , \tag{1}$$

where the expectation is taken over the randomness used by the protocol. Then, by the Markov inequality, the probability that the message is of size at least $100 \cdot s$ is at most $\frac{1}{100}$.

Given Π', we now construct a protocol Π'' with maximum message size $100 \cdot s$ that solves LP with slightly increased error: Whenever Π' sends a message of size at most $100 \cdot s$, Π'' also sends this message and the protocol behaves exactly the same as Π'. However, when Π' sends a message of size at least $100 \cdot s$, Π'' aborts and thus fails. Since the probability of sending a message of size larger than $100 \cdot s$ is $1/100$, the error probability of Π'' is $(\frac{1}{4} - \frac{1}{100}) + \frac{1}{100} = \frac{1}{4}$. From Theorem 13, we obtain that the communication cost of Π'' is $\Omega(r)$, which implies that $100 \cdot s = \Omega(r)$. Hence, combined with Inequality 1, we conclude that

$$\Omega(r) = I_{\mathcal{D}_{\mathrm{SLP}}(r)}(M \ : \ \Pi) \leq I_{\mathcal{D}_{\mathrm{SLP}}(r)}(M \ : \ \Pi \mid R) = \mathrm{ICost}_{\mathcal{D}_{\mathrm{SLP}}(r)}(\Pi) ,$$

where the inequality follows from property **P1** stated in the preliminaries. ◀

4.2 Hard Input Distribution and Direct Sum Argument

The input distribution $\mathcal{D}_{\mathrm{LP}}(n)$ used to obtain our main lower bound is stated in Figure 4.

We now argue that a protocol $\Pi_{\mathrm{LP}}(n)$ that solves LP under distribution $\mathcal{D}_{\mathrm{LP}}(n)$ can be used to obtain a protocol $\Pi_{\mathrm{SLP}}(r)$ that solves LP under $\mathcal{D}_{\mathrm{SLP}}(r)$, where r is the size of the induced matchings of the RS-graph used in $\mathcal{D}_{\mathrm{LP}}(n)$. This establishes a connection between the information costs of $\Pi_{\mathrm{LP}}(n)$ and $\Pi_{\mathrm{SLP}}(r)$. We use the reduction stated in Algorithm 2.

This reduction has the following properties:

Input Distribution $\mathcal{D}_{\text{LP}}(n)$:

Let $H = (A^H, B^H, E^H)$ be an (r,t)-RS graphs with $|A^H| = |B^H| = n$ and induced matchings $M_1^H, M_2^H, \ldots, M_t^H$.

Alice's Input: Edge set E_A

Given $H = (A^H, B^H, E^H)$, we construct the directed bipartite graph $G = (A, B = B_1 \dot\cup B_2, E)$, where:

- A is a copy of A^H, and B_1 and B_2 are copies of B^H.
- For every edge $e = \{a, b\} \in E^H$, we flip an unbiased coin $X_{a,b}$. If it comes out heads then the directed edge (a, b) between the sets A and B_1 is introduced, and if it comes out tail then the directed edge (a, b) between the sets A and B_2 is introduced.

We denote the edges in E that originated from the matching M_i^H, for any i, by M_i. Alice holds the edges $E_A = \cup_i M_i$.

Bob's Input: Edge set E_B

- Let $J \in [t]$ be a uniform random index.
- Consider the matching $M_J^H \in E_H$ and let $A'_H = A(M_J^H)$ and $B'_H = B(M_J^H)$. Let N_J be a uniform random matching between A' and B'.
- Bob introduces two copies of N_J into G: To this end, let $A' \subseteq A$ be the copy of A'_H in G, let $B'_1 \subseteq B_1$ be the copy of B' in B_1, and let $B'_2 \subseteq B_2$ be the copy of B' in B_2. The first copy N_J^1 is introduced between A' and B'_1, and the second copy N_J^2 is introduced between A' and B'_2. The edges in N_J^1 and N_J^2 are directed such that the A-vertex constitutes the head and the B-vertex the tail of the directed edge.

Bob holds the edges $E_B = N_J^1 \cup N_J^2$.

Figure 4 Input Distribution $\mathcal{D}_{\text{LP}}(n)$.

Algorithm 2 Construction of protocol $\Pi_{\text{SLP}}(r)$.

Require:

1. Protocol $\Pi_{\text{LP}}(n)$ that solves $\mathsf{LP}^{\alpha}_{1/4}$ under distribution $\mathcal{D}_{\text{LP}}(n)$
2. Input $(M, N^1, N^2) \sim \mathcal{D}_{\text{SLP}}(r)$, Alice holds matching M, Bob holds matchings N^1, N^2

1: Alice and Bob use public randomness to generate a uniform random index $J \in [t]$
2: Alice samples graph $G \sim \mathcal{D}_{\text{LP}}(n)$ and updates matching M_J in G such that $M_J = M$
3: Bob set $N_J^1 = N^1$ and $N_J^2 = N^2$ and adds these to G
4: Alice and Bob run the protocol $\Pi_{\text{LP}}(n)$ on G, denote by $\mathcal{P}$ the output of $\Pi_{\text{LP}}(n)$
5: **return** Longest sub-path of $\mathcal{P}$ that solely uses edges in $M_J \cup N_J^1 \cup N_J^2$

▶ **Lemma 15.** *Consider the reduction in Algorithm 2. Given any path $\mathcal{P}$ in G of length at least 2, the path $\mathcal{P}'$ with the first and last edge removed only uses edges from $M_J \cup N_J^1 \cup N_J^2$.*

Proof. Let $\mathcal{P}$ be a path of length at least 2 in G. Since G is bipartite, every other edge in $\mathcal{P}$ must be an edge from $N_J^1 \cup N_J^2$ since these are the only edges with head in A. Observe that the edges M_J are the only edges with both endpoints in $A(N_J^1 \cup N_J^2)$ and $B(N_J^1 \cup N_J^2)$. Hence, for any three consecutive edges e, f, g in path $\mathcal{P}$, if e and g are edges from $N_J^1 \cup N_J^2$, then f must be an edge from M_J. It follows that if an edge that is not contained in $M_J \cup N_J^1 \cup N_J^2$ is included in $\mathcal{P}$ then this edge must be the first or the last edge of $\mathcal{P}$. ◀

Given our reduction, we now relate the information costs of $\Pi_{\text{SLP}}(r)$ and $\Pi_{\text{LP}}(n)$:

▶ **Lemma 16.** *Let $\Pi_{LP}(n)$ be a protocol for LP^{α}_{ϵ}, for some parameters α and ϵ, and let $\Pi_{SLP}(r)$ be the protocol obtained from $\Pi_{LP}(n)$ via the reduction given in Algorithm 2. Then, $\Pi_{SLP}(r)$ has approximation factor $O(\alpha)$, errs with the same probability ϵ, and:*

$$\mathrm{ICost}_{\mathcal{D}_{LP}(n)}(\Pi_{LP}(n)) = t \cdot \mathrm{ICost}_{\mathcal{D}_{SLP}(r)}(\Pi_{SLP}(r)) \ .$$

Proof. We denote by R_{SLP} and R_{LP} the public randomness used in $\Pi_{\mathrm{SLP}}(r)$ and in $\Pi_{\mathrm{LP}}(n)$, respectively. Then (see the rules **P1**, ..., **P4** in the preliminaries):

$$\begin{aligned}
\mathrm{ICost}_{\mathcal{D}_{\mathrm{SLP}}(r)}(\Pi_{\mathrm{SLP}}(r)) &= I_{\mathcal{D}_{\mathrm{SLP}}(r)}(M \ : \ \Pi_{\mathrm{SLP}}(r) \mid R_{\mathrm{SLP}}) \\
&= I_{\mathcal{D}_{\mathrm{SLP}}(r)}(M_J \ : \ \Pi_{\mathrm{LP}(n)} \mid R_{\mathrm{LP}}, J) \\
&= \mathbb{E}_{j \leftarrow J} \, I_{\mathcal{D}_{\mathrm{LP}}(n)}(M_J \ : \ \Pi_{\mathrm{LP}(n)} \mid R_{\mathrm{LP}}, J = j) && \textbf{P2} \\
&= \frac{1}{t} \cdot \sum_{j \in [t]} I_{\mathcal{D}_{\mathrm{LP}}(n)}(M_j \ : \ \Pi_{\mathrm{LP}(n)} \mid R_{\mathrm{LP}}, J = j) \\
&= \frac{1}{t} \cdot \sum_{j \in [t]} I_{\mathcal{D}_{\mathrm{LP}}(n)}(M_j \ : \ \Pi_{\mathrm{LP}(n)} \mid R_{\mathrm{LP}}) && \textbf{P3} \\
&\leq \frac{1}{t} \cdot \sum_{j \in [t]} I_{\mathcal{D}_{\mathrm{LP}}(n)}(M_j \ : \ \Pi_{\mathrm{LP}(n)} \mid M_1, \ldots, M_{j-1}, R_{\mathrm{LP}}) && \textbf{P1} \\
&= \frac{1}{t} \cdot I_{\mathcal{D}_{\mathrm{LP}}(n)}(M_1, \ldots, M_t \ : \ \Pi_{\mathrm{LP}(n)} \mid R_{\mathrm{LP}}) && \textbf{P4} \\
&= \mathrm{ICost}_{\mathcal{D}_{\mathrm{LP}}(n)}(\Pi_{\mathrm{LP}}(n)) \ .
\end{aligned}$$

Regarding the approximation factor, observe that $\mathrm{lp}(H) \leq \mathrm{lp}(G)$. Furthermore, by Lemma 15, the path found by $\Pi_{\mathrm{SLP}}(r)$ is at most by 2 shorter than the path found by $\Pi_{\mathrm{LP}}(n)$, which in turn is of length at least $\mathrm{lp}(G)/\alpha$. Hence, the approximation factor of $\Pi_{\mathrm{SLP}}(r)$ is bounded by $\frac{\mathrm{lp}(H)}{\mathrm{lp}(G)/\alpha - 2} \leq \frac{\mathrm{lp}(G)}{\mathrm{lp}(G)/\alpha - 2} = O(\alpha)$. ◀

We are now ready to state our main lower bound result for directed graphs.

▶ **Theorem 17.** *The randomized one-way communication complexity of $LP^{O(r/\log r)}_{1/4-\frac{1}{500}}(n)$ is $\Omega(r \cdot t)$.*

Proof. Let $\Pi_{\mathrm{LP}}(n)$ be a protocol for $\mathsf{LP}^{O(r/\log r)}_{1/4-\frac{1}{500}}(n)$. Then, from Lemmas 16 and 14, we obtain

$$\mathrm{ICost}_{\mathcal{D}_{\mathrm{LP}}(n)}(\Pi_{\mathrm{LP}}(n)) = \Omega(r \cdot t) \ .$$

Since the choice of $\Pi_{\mathrm{LP}}(n)$ was arbitrary, and information complexity is a lower bound on communication complexity, the result follows. ◀

Finally using the well-known connection between streaming algorithms and one-way two-party communication protocols as well as the RS-graph construction by Alon et al. as stated in Theorem 7, we obtain the following theorem:

▶ **Theorem 3.** *Any one-pass algorithm for $LP^{O(n^{1-o(1)}/\log(n^{1-o(1)})}_{1/4-\frac{1}{500}}(n) = LP^{O(n^{1-o(1)})}_{1/4-\frac{1}{500}}(n)$ requires space $\Omega(n^2)$.*

5 Conclusion

We studied one-pass streaming algorithms for LP and showed that, in both insertion-only and insertion-deletion streams, for undirected graphs, there are semi-streaming algorithms that find paths of lengths at least $\frac{d}{3}$ with high probability, where d is the average degree of the input graph. The algorithm can also give an α-approximation algorithm that uses space $\tilde{O}(n^2/\alpha)$. We then showed that no such result can be obtained for directed graphs in that a $n^{1-o(1)}$-approximation requires space $\Omega(n^2)$, even in insertion-only streams. We also showed that semi-streaming algorithms in the insertion-only model for undirected graphs cannot yield an arbitrarily small constant factor approximation, and, in insertion-deletion streams, space $\Omega(n^2/\alpha^3)$ is necessary to obtain an α-approximation in undirected graphs.

We conclude with two open questions: First, while we resolved the space complexity of one-pass algorithms for directed graphs in both the insertion-only and the insertion-deletion models, the space complexity for undirected graphs, in particular, in the insertion-only model, remains wide open. Can we close this gap? Second, for undirected graphs, are there multi-pass semi-streaming algorithms that compute paths longer than $\Theta(d)$, where d is the average degree of the input graph?

References

1 Kook Jin Ahn, Graham Cormode, Sudipto Guha, Andrew McGregor, and Anthony Wirth. Correlation clustering in data streams. *Algorithmica*, 83(7):1980–2017, 2021. doi:10.1007/S00453-021-00816-9.

2 Noga Alon, Ankur Moitra, and Benny Sudakov. Nearly complete graphs decomposable into large induced matchings and their applications. In Howard J. Karloff and Toniann Pitassi, editors, *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19–22, 2012*, pages 1079–1090. ACM, 2012. doi:10.1145/2213977.2214074.

3 Sepehr Assadi. A two-pass (conditional) lower bound for semi-streaming maximum matching. In Joseph (Seffi) Naor and Niv Buchbinder, editors, *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9–12, 2022*, pages 708–742. SIAM, 2022. doi:10.1137/1.9781611977073.32.

4 Sepehr Assadi, Soheil Behnezhad, Christian Konrad, Kheeran K. Naidu, and Janani Sundaresan. Settling the pass complexity of approximate matchings in dynamic graph streams. In Yossi Azar and Debmalya Panigrahi, editors, *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2025, New Orleans, LA, USA, January 12-15, 2025*, pages 864–904. SIAM, 2025. doi:10.1137/1.9781611978322.25.

5 Sepehr Assadi, Aaron Bernstein, Zachary Langley, Lap Chi Lau, and Robert Wang. Streaming and communication complexity of load-balancing via matching contractors. In Yossi Azar and Debmalya Panigrahi, editors, *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2025, New Orleans, LA, USA, January 12-15, 2025*, pages 3423–3449. SIAM, 2025. doi:10.1137/1.9781611978322.113.

6 Sepehr Assadi, Yu Chen, and Sanjeev Khanna. Sublinear algorithms for $(\Delta + 1)$ vertex coloring. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 767–786. SIAM, 2019. doi:10.1137/1.9781611975482.48.

7 Sepehr Assadi, Sanjeev Khanna, and Yang Li. On estimating maximum matching size in graph streams. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1723–1742. SIAM, 2017. doi:10.1137/1.9781611974782.113.

8 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.

9 Sepehr Assadi, Christian Konrad, Kheeran K. Naidu, and Janani Sundaresan. O(log log n) passes is optimal for semi-streaming maximal independent set. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 847–858. ACM, 2024. doi:10.1145/3618260.3649763.

10 Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto. Narrow sieves for parameterized paths and packings. *Journal of Computer and System Sciences*, 87:119–139, 2017. doi:10.1016/j.jcss.2017.03.003.

11 Andreas Björklund, Thore Husfeldt, and Sanjeev Khanna. Approximating longest directed paths and cycles. In Josep Díaz, Juhani Karhumäki, Arto Lepistö, and Donald Sannella, editors, *Automata, Languages and Programming*, pages 222–233, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg. doi:10.1007/978-3-540-27836-8_21.

12 Amit Chakrabarti, Prantar Ghosh, Andrew McGregor, and Sofya Vorotnikova. Vertex ordering problems in directed graph 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 1786–1802. SIAM, 2020. doi:10.1137/1.9781611975994.109.

13 Yi-Jun Chang, Martin Farach-Colton, Tsan-sheng Hsu, and Meng-Tsung Tsai. Streaming complexity of spanning tree computation. In Christophe Paul and Markus Bläser, editors, *37th International Symposium on Theoretical Aspects of Computer Science, STACS 2020, March 10-13, 2020, Montpellier, France*, volume 154 of *LIPIcs*, pages 34:1–34:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPICS.STACS.2020.34.

14 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.

15 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.

16 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.

17 Eldar Fischer, Eric Lehman, Ilan Newman, Sofya Raskhodnikova, Ronitt Rubinfeld, and Alex Samorodnitsky. Monotonicity testing over general poset domains. In John H. Reif, editor, *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 474–483. ACM, 2002. doi:10.1145/509907.509977.

18 Ashish Goel, Michael Kapralov, and Sanjeev Khanna. On the communication and streaming complexity of maximum bipartite matching. 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 468–485. SIAM, 2012. doi:10.1137/1.9781611973099.41.

19 S. W. Golomb. Random permutations. *Bull. Amer. Math. Soc. 70*, 1964.

20 S. W. Golomb, L. R. Welch, and R. M. Goldstein. Cycles from nonlinear shift registers. Technical report, Progress Rep. No. 20-389, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California, 1959.

21 Prahladh Harsha, Rahul Jain, David A. McAllester, and Jaikumar Radhakrishnan. The communication complexity of correlation. In *22nd Annual IEEE Conference on Computational Complexity (CCC 2007), 13-16 June 2007, San Diego, California, USA*, pages 10–23. IEEE Computer Society, 2007. doi:10.1109/CCC.2007.32.

22 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.

23 Ramkumar G. D. S. Karger D, Motwani R. On approximating the longest path in a graph. *Algorithmica*, pages 82–98, 1997. doi:10.1007/BF02523689.

24 Shahbaz Khan and Shashank K. Mehta. Depth First Search in the Semi-streaming Model. In Rolf Niedermeier and Christophe Paul, editors, *36th International Symposium on Theoretical Aspects of Computer Science (STACS 2019)*, volume 126 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 42:1–42:16, Dagstuhl, Germany, 2019. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.STACS.2019.42.

25 Lasse Kliemann, Christian Schielke, and Anand Srivastav. A streaming algorithm for the undirected longest path problem. In Piotr Sankowski and Christos D. Zaroliagis, editors, *24th Annual European Symposium on Algorithms, ESA 2016, August 22-24, 2016, Aarhus, Denmark*, volume 57 of *LIPIcs*, pages 56:1–56:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPICS.ESA.2016.56.

26 Christian Konrad and Kheeran K. Naidu. On two-pass streaming algorithms for maximum bipartite matching. In Mary Wootters and Laura Sanità, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2021, August 16-18, 2021, University of Washington, Seattle, Washington, USA (Virtual Conference)*, volume 207 of *LIPIcs*, pages 19:1–19:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.APPROX/RANDOM.2021.19.

27 Christian Konrad and Kheeran K. Naidu. An unconditional lower bound for two-pass streaming algorithms for maximum matching approximation. In David P. Woodruff, editor, *Proceedings of the 2024 ACM-SIAM Symposium on Discrete Algorithms, SODA 2024, Alexandria, VA, USA, January 7-10, 2024*, pages 2881–2899. SIAM, 2024. doi:10.1137/1.9781611977912.102.

28 Christian Konrad, Kheeran K. Naidu, and Arun Steward. Maximum matching via maximal matching queries. In Petra Berenbrink, Patricia Bouyer, Anuj Dawar, and Mamadou Moustapha Kanté, editors, *40th International Symposium on Theoretical Aspects of Computer Science, STACS 2023, March 7-9, 2023, Hamburg, Germany*, volume 254 of *LIPIcs*, pages 41:1–41:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICS.STACS.2023.41.

29 Andrew McGregor. Finding graph matchings in data streams. In Chandra Chekuri, Klaus Jansen, José D. P. Rolim, and Luca Trevisan, editors, *Approximation, Randomization and Combinatorial Optimization, Algorithms and Techniques, 8th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, APPROX 2005 and 9th InternationalWorkshop on Randomization and Computation, RANDOM 2005, Berkeley, CA, USA, August 22-24, 2005, Proceedings*, volume 3624 of *Lecture Notes in Computer Science*, pages 170–181. Springer, 2005. doi:10.1007/11538462_15.

30 Slobodan Mitrovi?, Anish Mukherjee, Piotr Sankowski, and Wen-Horng Sheu. Faster semi-streaming matchings via alternating trees, 2025. doi:10.48550/arXiv.2412.19057.

31 Anup Rao and Amir Yehudayoff. *Communication Complexity: and Applications*. Cambridge University Press, 2020.

32 Jeffrey Scott Vitter. Random sampling with a reservoir. *ACM Trans. Math. Softw.*, 11(1):37–57, 1985. doi:10.1145/3147.3165.

A Technical Lemmas

▶ **Lemma 18.** *Let a, b, c be positive with $a - c \geq b$. Then:* $\frac{\binom{a-c}{b}}{\binom{a}{b}} \leq \exp(-\frac{b \cdot c}{a})$.

Proof. We compute as follows:

$$\frac{\binom{a-c}{b}}{\binom{a}{b}} = \frac{\frac{(a-c)!}{(a-b-c)!b!}}{\frac{a!}{(a-b)!b!}} = \frac{(a-c)\cdot(a-c-1)\cdot\ldots\cdot(a-b-c+1)}{a\cdot(a-1)\cdot\ldots\cdot(a-b+1)} \leq (1-\frac{c}{a})^b \leq \exp(-\frac{bc}{a}) ,$$

where we used the inequality $1 + x \leq \exp(x)$, which holds for all x. ◀

▶ **Lemma 19.** *Let $G = (V, E)$ be a graph with $|V| = n$, $|E| = m$, and average degree $d = 2\frac{m}{n}$. Then, there exists a subset of vertices $U \subseteq V$ such that the vertex-induced subgraph $G[U]$ has minimum degree greater than $\frac{d}{2}$.*

Proof. We iteratively remove vertices of degree at most $d/2$ from G until no such vertex is left. Let G_i be the graph with the first i vertices removed, and let $G_0 = G$. We denote m_i the number of edges in G_i, and $n_i = n - i$ the number of vertices in G_i. It can then be seen by induction that the average degree d_i of every graph G_i is still at least $d_0 = d$. Indeed, observe that removing a vertex of degree at most $d/2$ removes at most $d/2$ edges from the graph. Then, the average degree of graph G_{i+1} is:

$$2 \cdot \frac{m_{i+1}}{n_{i+1}} \geq 2 \cdot \frac{m_i - \frac{d}{2}}{n-(i+1)} = 2 \cdot \frac{\frac{d}{2}(n-i) - \frac{d}{2}}{n-(i+1)} = d .$$

Then, since the average degree remains as high as d throughout, and we only ever remove vertices of degree at most $d/2$, the process must leave a non-empty graph with minimum degree at least $\lfloor d/2 + 1 \rfloor$ behind. The set U then is the set of vertices that are not removed by this process. ◀