Tight Bounds for Graph Problems in Insertion Streams*

Xiaoming Sun[1] and David P. Woodruff[2]

1 **Institute of Computing Technology, CAS, China**
sunxiaoming@ict.ac.cn
2 **IBM Almaden, USA**
dpwoodru@us.ibm.com

Abstract

Despite the large amount of work on solving graph problems in the data stream model, there do not exist tight space bounds for almost any of them, even in a stream with only edge insertions. For example, for testing connectivity, the upper bound is $O(n \log n)$ bits, while the lower bound is only $\Omega(n)$ bits. We remedy this situation by providing the first tight $\Omega(n \log n)$ space lower bounds for randomized algorithms which succeed with constant probability in a stream of edge insertions for a number of graph problems. Our lower bounds apply to testing bipartiteness, connectivity, cycle-freeness, whether a graph is Eulerian, planarity, H-minor freeness, finding a minimum spanning tree of a connected graph, and testing if the diameter of a sparse graph is constant. We also give the first $\Omega(nk \log n)$ space lower bounds for deterministic algorithms for k-edge connectivity and k-vertex connectivity; these are optimal in light of known deterministic upper bounds (for k-vertex connectivity we also need to allow edge duplications, which known upper bounds allow). Finally, we give an $\Omega(n \log^2 n)$ lower bound for randomized algorithms approximating the minimum cut up to a constant factor with constant probability in a graph with integer weights between 1 and n, presented as a stream of insertions and deletions to its edges. This lower bound also holds for cut sparsifiers, and gives the first separation of maintaining a sparsifier in the data stream model versus the offline model.

1998 ACM Subject Classification F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases communication complexity, data streams, graphs, space complexity

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2015.435

1 Introduction

In a data stream one sees a sequence of elements $a_1, \ldots, a_m$ one by one and one would like to evaluate certain functions of the stream. There are example data streams which come from internet search logs, network traffic, sensor networks, and scientific data streams. The elements a_i may be numbers, points, edges in a graph, etc. Due to the sheer size of the sequence, very stringent requirements are imposed on a data stream algorithm. For instance, it is often assumed that the algorithm can only make one, or a small number, of passes over the stream. Moreover, the algorithm is assumed to have very limited memory, which in particular makes storing the stream in its entirety infeasible. We refer the reader to the surveys [4, 29] for a more thorough introdution to this area.

* Xiaoming Sun was supported in part by the National Natural Science Foundation of China Grant 61170062, 61222202, 61433014 and the China National Program for support of Top-notch Young Professionals.

18th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'15) / 19th Int'l Workshop on Randomization and Computation (RANDOM'15).
Editors: Naveen Garg, Klaus Jansen, Anup Rao, and José D. P. Rolim; pp. 435–448

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

In this paper we focus on the case when the elements a_i are edges of an underlying graph G. That is, we insert edges into the graph one at a time, and would like to compute a function of G. Graphs arise is many applications to model relationships bewteen basic entities, such as links between webpages or network flows between destinations. These graphs are often massive and running classical algorithms on such graphs has proven quite challenging. This has motivated the recent work on processing graphs in the data stream model. In the case when only edges are inserted into the stream (as opposed to also being deleted), we have algorithms for a number of problems, including testing connectivity, finding minimum spanning trees, computing cut and spectral sparsifiers, counting subgraphs, finding matchings, and many other problems. We refer the reader to the survey by McGregor [28] for an overview of these results.

It is known for many graph problems that there is a space lower bound of $\Omega(n)$ [16, 17]. The graph streaming model is therefore sometimes identified with the "semi-streaming" model, which allows the streaming algorithm to use $n \cdot \text{polylog}(n)$ bits of space. Note that this is still a substantial improvement over the naïve algorithm of storing the graph, which may take $\Omega(n^2)$ bits of space. Despite the fact that we have $n \cdot \text{polylog}(n)$ space upper bounds and $\Omega(n)$ space lower bounds for a number of graph problems in the data stream model, we are not aware of a single natural problem for which we have asymptotically tight bounds. Could it be that simple upper bounds, such as the $O(n \log n)$ bit upper bound for testing if a graph is connected by maintaining a spanning forest in the stream, can be improved using more clever hashing techniques to represent the edges, perhaps chosen adaptively as the stream is presented? Such a scheme could potentially allow us to avoid spending $O(\log n)$ bits to remember each edge in the spanning forest.

Dowling and Wilson [14] show that the deterministic communication complexity of connectivity is $\Omega(n \log n)$ bits, see also [31] for a discussion. Via standard connections to data streams (see Section 1.3 below), this implies any deterministic streaming algorithm requires $\Omega(n \log n)$ bits. However, to the best of our knowledge, there was no lower bound for randomized 1-way protocols known, prior to this work, stronger than $\Omega(n)$ bits.

Graph problems in a data stream should be contrasted to a number of other areas in streaming for which tight asymptotic space bounds are known, such as estimating frequency moments [2, 7, 18, 22, 23, 27], empirical entropy [8, 11, 20, 21, 22], numerical linear algebra [10], and compressed sensing [3, 30]. The goal of this paper is to remedy this situation.

1.1 Our Results

Throughout this paper we will restrict our attention to 1-pass algorithms and focus on their space complexity. Our focus is on the model in which edges are only allowed to be inserted into the graph, i.e., the "insertion model", rather than also being allowed to be deleted, which is referred to as the "turnstile model". Since we prove lower bounds, this only makes our bounds stronger. We will, however, show how to use our techniques to obtain stronger lower bounds in the turnstile model for approximating the minimum cut of a graph.

Our results are summarized in Table 1. We provide the first tight space lower bounds for a number of graph problems in a stream of edge insertions. In particular, for randomized algorithms which succeed with constant probability, we show an $\Omega(n \log n)$ lower bound for testing if a graph is connected, testing if a graph with $O(n)$ edges has diameter at most 5 or diameter ∞, testing if a graph is Eulerian, testing if a graph is bipartite, testing if a graph is cycle-free, finding a minimum spanning tree in a graph that is promised to be connected, testing if a graph is planar, and testing whether a graph contains a fixed graph H as a minor. No lower bounds better than $\Omega(n)$ were known for any of these problems. Many of the upper

Table 1 Summary of our results. All lower bounds are new and given by this work. In the comments we say "UB stores graph" for those problems for which there is no graph with more than $C \cdot n$ edges, for a constant $C > 0$, satisfying the property. For such problems it suffices to store all edges in the graph and abort if more than $C \cdot n$ edges are inserted, yielding an $O(n \log n)$ bit upper bound. All upper bounds, with the exception of Eulerian-testing, either come from previous work or "UB stores graph" applies to the problem. For Eulerian-testing the upper bound maintains a spanning forest and the parities of node degrees, using that a graph is Eulerian iff it is connected and all node degrees are even.

Problem	Lower Bound	Upper Bound	Comments
Connectivity	$\Omega(n \log n)$	$O(n \log n)$ [28]	–
Diameter in sparse graphs	$\Omega(n \log n)$	$O(n \log n)$	UB stores graph
Eulerian-testing	$\Omega(n \log n)$	$O(n \log n)$	UB: spanning forest, degrees
Bipartiteness	$\Omega(n \log n)$	$O(n \log n)$ [28]	–
Cycle-freeness	$\Omega(n \log n)$	$O(n \log n)$	UB stores graph
MST in connected graphs	$\Omega(n \log n)$	$O(n \log n)$ [28]	–
Planarity	$\Omega(n \log n)$	$O(n \log n)$	UB stores graph
H-Minor Free	$\Omega(n \log n)$	$O(n \log n)$	UB stores graph
k-Edge Connectivity	$\Omega(kn \log n)$	$O(kn \log n)$ [13]	Deterministic bounds
k-Vertex Connectivity w/edge duplications	$\Omega(kn \log n)$	$O(kn \log n)$ [15]	Deterministic bounds
$O(1)$-Approximate Minimum Cut	$\Omega(n \log^2 n)$	$O(n \log^4 n)$ [1, 24]	Bounds in the turnstile model

bounds follow simply by storing all edges in the graph and aborting if the number of edges is too large; see Table 1 for details.

Next, we turn to k-edge connectivity, which is equivalent to testing if the minimum cut of the graph is at least k. There is a deterministic space upper bound of $O(kn \log n)$ bits [13, 28]. We show a matching $\Omega(kn \log n)$ bit lower bound for deterministic algorithms. For randomized algorithms our space bound is a weaker $\Omega(kn)$, and closing the $\log n$ factor gap for deterministic and randomized k-edge connectivity algorithms remains an important open question. For k-vertex connectivity, there is a deterministic space upper bound of $O(kn \log n)$ bits due to [15]. We are able to prove a matching $\Omega(kn \log n)$ bit lower bound for deterministic algorithms, but require that multiple edges are allowed, i.e., our hard instance is a multi-graph for this problem. We notice, however, that the upper bound of [15] also holds for multi-graphs. Our lower bound becomes a weaker $\Omega(kn)$ in the case of randomized algorithms, and closing the $\log n$ factor gap for randomized algorithms for k-vertex connectivity and/or removing the edge duplication assumption is an important open problem.

Finally, we illustrate the power of our technique by proving an $\Omega(n \log^2 n)$ lower bound for approximating the minimum cut up to a constant factor of a graph with integer weights between 1 and n, in the turnstile model. The same lower bound holds for cut sparsifiers (since they can be used to approximate the minimum cut), and gives the first separation of maintaining a sparsifier in the data stream model versus the offline model. Indeed, in the offline model, by a result of Batson et al. [6] it is possible to build a cut sparsifier (in fact, a stronger notion of a spectral sparsifier) of a graph using only $O(n)$ reweighted edges of the input graph, with $O(\log n)$ bits to specify each edge and its weight. Our $\Omega(n \log^2 n)$ bit lower bound shows it is fundamentally impossible to implement the algorithm of [6] in a dynamic stream. For general integer edge weights between 1 and W, our lower bound is $\Omega(n \log n \log W)$.

1.2 Our Techniques

Our results come from identifying a new two-player one-way communication problem which generalizes the well-studied Index problem [26], to a problem Perm which is more suitable for proving graph lower bounds. Despite the simplicity of the Perm problem, we are able to apply it to the wide array of problems above. In this problem, Alice is given a permutation σ on $[n] \stackrel{\text{def}}{=} \{1, 2, \ldots, n\}$, which she represents in a slightly redundant way as an $n \log n$-length bitstring $\sigma(1), \ldots, \sigma(n)$ formed by concatenating the images of $1, 2, \ldots, n$ under σ. We call this the *redundant encoding* of σ. Bob is interested in obtaining the i-th bit in the redundant encoding of σ. We show that if Alice sends a single message to Bob, then for Bob to succeed with constant probability, Alice's message needs to be $\Omega(n \log n)$ bits long. In other words, the randomized 1-way communication complexity $R^{1-way}(\mathsf{Perm}) = \Omega(n \log n)$. We generalize this to the case where Alice has r permutations $\sigma^1, \ldots, \sigma^r$ each on $[n]$, while Bob now has an index $i \in [n]$, an index $k \in [r]$, as well as Alice's permutations $\sigma^{k+1}, \ldots, \sigma^r$, and Bob is interested in the i-th bit in the redundant encoding of σ^k. We call this problem r-AugmentedPerm and show $R^{1-way}(r\text{-}\mathsf{AugmentedPerm}) = \Omega(rn \log n)$.

After identifying Perm and r-AugmentedPerm as the right problems to study, the proofs of their respective lower bounds follow standard information-theoretic arguments used to prove lower bounds for Index and direct sum theorems in streaming [5, 9], with small modifications to account for the redundancy. The second part of our proofs is reducing graph problems to these communication problems. The core idea of our lower bounds is to identify a permutation σ as a random matching on a bipartite graph with n vertices in the left part L and n vertices in the right part R. This is Alice's input graph G in many of our reductions. Alice runs the streaming algorithm on G, sends the state to Bob, who then inserts edges into G in a problem-specific way. As Bob is interested in learning a bit of the redundant encoding of Alice's permutation, this corresponds to a bit j of the unique neighbor in R of a vertex $u \in L$. We therefore create gadgets which group all vertices in R into two groups, based on the value of their j-th bit, and connect these groups to vertices in L in different ways depending on the particular problem.

1.3 Preliminaries

Let $f : X \times Y \to \{0, 1\}$ be a Boolean function, where X and Y are two arbitrary sets. In the *one-way* communication model Alice receives an input $x \in X$ and Bob receives an input $y \in Y$. Alice is only allowed to send one message to Bob and no message is allowed to be sent from Bob to Alice. The goal is for Bob to compute $f(x, y)$. The communication cost is measured by the number of bits Alice sends in the worst case. Denote by $D^{1-way}(f)$ the minimum communication cost over all deterministic one-way protocols for f. For a randomized protocol P, we say P has error probability at most ϵ if $\Pr(P(x, y) = f(x, y)) \geq 1 - \epsilon$ for all inputs x and y. The randomness here is only over the private coin tosses of Alice and Bob. The one-way (bounded-error) randomized communication complexity of f, denote by $R^{1-way}(f)$, is the minimum communication cost over all randomized one-way protocols for f with error probability at most $1/3$.

Communication lower bounds on $D^{1-way}(f)$ and $R^{1-way}(f)$ provide lower bounds on the memory required of deterministic and randomized data stream algorithms, respectively, via a standard reduction. Indeed, Alice creates a stream σ_x from her input x, and runs the streaming algorithm on σ_x, passing the state of the algorithm to Bob. Bob creates a stream σ_y from his input y, and continues the execution of the streaming algorithm on σ_y. If the output of the streaming algorithm on the concatenated stream $\sigma_x \circ \sigma_y$ can be used

to solve the problem f, either deterministically or with constant error probability, then the space complexity of the streaming algorithm must be at least $D^{1-way}(f)$ or $R^{1-way}(f)$, respectively.

We also need a few concepts and notation from information theory. We refer the reader to [12] for a more comprehensive introduction. We give a short primer on the standard properties we use in Appendix A.

2 Permutation Problems

We consider the following communication problem Perm which will be used in our reductions. In this problem Alice is given a permutation σ of $[n]$, represented as an ordered list $\sigma(1), \sigma(2), \ldots, \sigma(n)$. This list has $n \log n$ bits. Bob is given an index $i \in [n \log n]$ and would like to know the i-th bit of σ. This problem is similar to the well-studied Index problem in randomized 1-way communication complexity, but it is slightly different in that $\sigma(1), \ldots, \sigma(n)$ is a redundant encoding of a permutation.

▶ **Lemma 1.** $R^{1-way}(\mathsf{Perm}) = \Omega(n \log n)$.

Proof. Let us place the uniform distribution on strings σ. Let $M(\sigma)$ be Alice's message to Bob, which is a random variable depending on the randomness of σ and the private random coin tosses of Alice. Then $R^{1-way}(\mathsf{Perm}) \geq H(M(\sigma)) \geq I(M(\sigma); \sigma)$, so it suffices to lower bound $I(M(\sigma); \sigma)$. We write σ_j to denote the j-th bit in the list $\sigma(1), \ldots, \sigma(n)$, where $j \in \{1, 2, \ldots, n \log n\}$.

By the chain rule,

$$\begin{aligned} I(M(\sigma); \sigma) &= \sum_{j=1}^{n \log n} I(M(\sigma); \sigma_j \mid \sigma_{<j}) \\ &= \sum_j (H(\sigma_j \mid \sigma_{<j}) - H(\sigma_j \mid M(\sigma), \sigma_{<j})) \\ &\geq \sum_j H(\sigma_j \mid \sigma_{<j}) - \sum_j H(\sigma_j \mid M(\sigma)) \\ &= H(\sigma) - \sum_j H(\sigma_j \mid M(\sigma)). \end{aligned}$$

Using Stirling's approximation, $H(\sigma) = \log n! = n \log(n/e) + O(\log n)$. Now consider $H(\sigma_j \mid M(\sigma))$. Since M is randomized protocol which succeeds on every pair of inputs (σ, i) with probability at least $9/10$, and M does not depend on j, it follows that from $M(\sigma)$ Bob can predict σ_i for any given i with probability at least $9/10$. By Fano's inequality, for each j this implies $H(\sigma_j \mid M(\sigma)) \leq H(1/10)$. Hence,

$$I(M(\sigma); \sigma) \geq n \log(n/e) - H(1/10) n \log n \geq (1 - H(1/10)) n \log n - O(n).$$

This completes the proof. ◀

We also define the problem r-AugmentedPerm, used in our reductions. In this problem, Alice is given r permutations $\sigma^1, \ldots, \sigma^r$, where each σ^j is represented as a list of $n \log n$ bits. Bob is given an index $i \in [n \log n]$, an index $k \in [r]$, and is given $\sigma^{k+1}, \sigma^{k+2}, \ldots, \sigma^r$. Bob's goal is to output σ^k_i, which is the i-th bit of σ^k.

▶ **Lemma 2.** $R^{1-way}(r\text{-}\mathsf{AugmentedPerm}) = \Omega(rn \log n)$.

Proof. As in the proof of Lemma 1, we place the uniform distribution on strings σ^j, for each $j \in [r]$, and the σ^j are independent. Let $M(\sigma^1, \ldots, \sigma^r)$ be Alice's message to Bob, which is a random variable depending on the randomness of $\sigma^1, \ldots, \sigma^r$ and her private random coin tosses. Then $R^{1-way}(r\text{-}\mathsf{AugmentedPerm}) \geq H(M(\sigma^1, \ldots, \sigma^r)) \geq I(M(\sigma^1, \ldots, \sigma^r); \sigma^1, \ldots, \sigma^r)$, so it suffices to lower bound $I(M(\sigma^1, \ldots, \sigma^r); \sigma^1, \ldots, \sigma^r)$.

By the chain rule,

$$I(M(\sigma^1, \ldots, \sigma^r); \sigma^1, \ldots, \sigma^r) = \sum_{k=1}^{r} I(M(\sigma^1, \ldots, \sigma^r); \sigma^k \mid \sigma^{k+1}, \ldots, \sigma^r) \quad (1)$$

We claim that for each $k \in [r]$,

$$I(M(\sigma^1, \ldots, \sigma^r); \sigma^k \mid \sigma^{k+1}, \ldots, \sigma^r) = \Omega(n \log n).$$

To see this, consider any fixing of the random variables $\sigma^{k+1}, \ldots, \sigma^r$, and let Π be a randomized protocol which succeeds on every input to $\mathsf{AugmentedPerm}$ with probability at least 9/10, over its random coin tosses. Then, given an input (σ, i) to the Perm problem, Alice and Bob can use Π as follows. Alice hardwires the fixed values of $\sigma^{k+1}, \ldots, \sigma^r$. Alice also sets $\sigma^k = \sigma$. Finally, she randomly and independently samples uniform permutations $\sigma^1, \ldots, \sigma^{k-1}$. Bob, given i as the input to Perm, also holds the input k and has the hardwired values of $\sigma^{k+1}, \ldots, \sigma^r$. Alice and Bob run Π on these inputs to $\mathsf{AugmentedPerm}$, and output whatever Π outputs. By correctness of Π, it follows that this is a correct 1-way protocol for the Perm problem with probability at least 9/10. Hence, as argued in the proof of Lemma 1, $I(M'(\sigma); \sigma) = \Omega(n \log n)$, where M' is Alice's resulting message function in the created protocol for Perm. By construction,

$$I(M(\sigma^1, \ldots, \sigma^r); \sigma^k \mid \sigma^{k+1}, \ldots, \sigma^r) = I(M'(\sigma); \sigma) = \Omega(n \log n),$$

as claimed. Plugging into (1), it follows that

$$I(M(\sigma^1, \ldots, \sigma^r); \sigma^1, \ldots, \sigma^r) = \Omega(rn \log n),$$

which completes the proof. ◀

3 Lower Bounds for Graph Problems

3.1 Connectivity

We start with an $\Omega(n \log n)$ bit lower bound for the randomized one-way communication of the graph connectivity problem, denoted Conn. In this problem, Alice has a subset E_A of edges of an undirected graph G on a set V of n vertices, while Bob has a disjoint subset E_B of the edges of G. Alice sends a single randomized message $M(E_A)$ to Bob, who should decide if the graph $(V, E_A \cup E_B)$ is connected. Bob should succeed with probability at least 9/10. We let $R^{1-way}(\mathsf{Conn})$ denote the minimum, over all correct protocol for Conn with probability at least 9/10, of the maximum length message sent, over all inputs and random coin tosses.

▶ **Theorem 3.** $R^{1-way}(\mathsf{Conn}) = \Omega(n \log n)$.

Proof. We perform a reduction from Perm on instances of size $n/2$. Alice, given a permutation σ, creates a perfect matching from $[n/2]$ to $[n/2]$ where the i-th left vertex connects to the $\sigma(i)$-th right vertex. Alice's edgeset E_A consists of the edges in this perfect matching. Let L and R denote the two parts of the vertex set V, each of size $n/2$.

Suppose Bob has the input i to Perm. This corresponds to the ℓ-th bit in $\sigma(j)$ for some $j \in [n/2]$ and $\ell \in [\log(n/2)]$. Bob creates his input edgeset to Conn from i as follows. Let $S \subset R$ denote the subset of vertices whose ℓ-th bit is equal to 0. Bob's input edgeset E_B consists of a spanning tree on the vertices in $(L \setminus \{j\}) \cup S$. We can ensure the edges of the spanning tree are disjoint from E_A by including a new vertex w, and including edges from all vertices in $(L \setminus \{j\}) \cup S$ to w.

Observe that since the vertices in $L \setminus \{j\}$ are connected, it follows that since we placed a perfect matching from L to R, that any vertex u is connected to any other vertex except possibly to j or $\sigma(j)$. Now, if the $\sigma(j)$-th right vertex has its ℓ-th bit equal to 0, then $\sigma(j)$ is connected to S, and hence to $L \setminus \{j\}$. It follows that the graph is connected. On the other hand, if the $\sigma(j)$-th right vertex has its ℓ-th bit equal to 1, then the edge from the j-th left vertex to the $\sigma(j)$-th right vertex is isolated, that is, it is not incident to any other vertices. In this case the graph is disconnected.

Let $M(E_A)$ be Alice's message to Bob in a protocol for Conn. Suppose Bob can decide, from $M(E_A)$ and E_B, if the resulting graph on vertex set $L \cup R$ and edgeset $E_A \cup E_B$ is connected with probability at least 9/10. It follows that Bob can solve Perm with probability at least 9/10, and therefore from Lemma 1, $R^{1-way}(\mathsf{Conn}) = \Omega(n \log n)$. ◀

▶ Remark. The lower bound in Theorem 3 is matched by a simple $O(n \log n)$ bit upper bound in which Alice sends a spanning forest of her edges to Bob.

3.2 Diameter

As a corollary of Conn, we show a lower bound for the following Diameter-k problem on sparse graphs, i.e., graphs with $O(n)$ edges: Given $k \in [n-1]$, Bob wants to decide if the diameter d of $(V, E_A \cup E_B)$ is at most k, or ∞.

▶ **Theorem 4.** *For any $k \geq 4$, $R^{1-way}(\mathsf{diameter}\text{-}k) = \Omega(n \log n)$.*

Proof. In the Conn proof, instead of only putting a spanning tree on the vertices in $(L \setminus \{j\}) \cup S$, we also put a clique on the vertices in $L \setminus \{j\}$ and a clique on the vertices in S. It follows that the diameter of $(V, E_A \cup E_B)$ is either $+\infty$ if the graph is disconnected, or 4 if the graph is connected. Therefore, the Diameter-k problem is as hard as Conn. ◀

▶ Remark. For sparse graphs the upper bound is just to store the entire graph with $O(n)$ edges.

3.3 Eulerian-Testing

In this part we show a lower bound for the Eulerian problem: Bob wants to decide if $(V, E_A \cup E_B)$ is an Eulerian graph.

▶ **Theorem 5.** $R^{1-way}(\mathsf{Eulerian}) = \Omega(n \log n)$.

Proof. In the Conn proof, call the graph $G_1 = (L, R, E)$, make another copy of the graph $G_2 = (L', R', E')$, i.e., in G_2 the edges are the same as in G_1. Alice and Bob also add the following edges to E_A and E_B: if there is an edge (u, v) in G_1, add edges (u, v') and (u', v). Let $V = L \cup R \cup L' \cup R'$. It is easy to check that the degree of every vertex is twice its degree in G_1, thus is an even number. Therefore, the resulting graph is Eulerian if and only if it is connected, but this is equivalent to the connectivity of G_1. Hence, $R^{1-way}(\mathsf{Eulerian}) = \Omega(n \log n)$. ◀

▶ Remark. An upper bound is to maintain a spanning forest to test connectivity, as well as to maintain the parities of all node degrees. Then one uses that a graph is Eulerian if and only if it is connected and all node degrees are even.

3.4 Bipartiteness

We now give a lower bound for the the Bipartite problem. In this problem Alice has a subset E_A of edges of an undirected graph G on a set V of n vertices, while Bob has a disjoint subset E_B of the edges of G. Alice sends a single randomized message $M(E_A)$ to Bob, who should decide if the graph $(V, E_A \cup E_B)$ is bipartite.

▶ **Theorem 6.** $R^{1-way}(\text{Bipartite}) = \Omega(n \log n)$.

Proof. We again reduce from Perm on instances of size $n/2$. Alice, given a permutation σ, creates a perfect matching from $[n/2]$ to $[n/2]$ where the i-th left vertex connects to the $\sigma(i)$-th right vertex. Alice's edgeset E_A consists of the edges in this perfect matching. Let L and R denote the two parts of the vertex set V, each of size $n/2$.

Suppose Bob has the input i to Perm. This corresponds to the ℓ-th bit in $\sigma(j)$ for some $j \in [n/2]$ and $\ell \in [\log(n/2)]$. Bob creates his input edgeset to Bipartite from i as follows. We create a new node w (so the input graph has $n+1$ nodes) and Bob includes an edge in E_B from the j-th vertex in L, denoted v, to w. Bob also includes all edges in E_B from w to any vertex in R whose ℓ-th bit is equal to 0.

Since E_A is a perfect matching, it is bipartite. Further, all edges in E_B are incident to w, and therefore G is bipartite if and only if there is no odd cycle which contains w. If we remove the edge $\{v, w\}$ then the graph is acyclic, and so any cycle must contain $\{v, w\}$, and hence also $\{v, \sigma(v)\}$, and hence also $\{\sigma(v), w\}$. It follows that an odd cycle exists iff $\{\sigma(v), w\}$ is in E_B, that is, iff the ℓ-th bit of $\sigma(j)$ is equal to 0.

It follows that Bob can solve Perm with probability at least 9/10, and therefore from Lemma 1, $R^{1-way}(\text{Bipartite}) = \Omega(n \log n)$. ◀

▶ Remark. There is an upper bound of $O(n \log n)$ bits for bipartiteness; see section 3.1 of [28]. It is stated as a streaming algorithm which immediately gives rise to a 1-way communication protocol.

3.5 Cycle-free

As a corollary of Bipartite, we show a lower bound for the following Cycle-free problem: Bob wants to decide if there is a cycle in $(V, E_A \cup E_B)$.

▶ **Theorem 7.** $R^{1-way}(\text{cycle-free}) = \Omega(n \log n)$.

Proof. In the Bipartite proof, if the ℓ-th bit of $\sigma(j)$ is 0, then there is a cycle between j, $\sigma(j)$ and w. If the ℓ-th bit of $\sigma(j)$ is 0, then there is no cycle. Therefore, $R^{1-way}(\text{cycle-free}) = \Omega(n \log n)$. ◀

▶ Remark. There is an upper bound of $O(n \log n)$ bits by storing the first $n-1$ edges of G. If G has more than $n-1$ edges, it necessarily contains a cycle. If it has fewer, one can test whether it contains a cycle.

3.6 Minimum Spanning Tree

We now present an application to the minimum spanning tree (MST) of a connected graph. In the MST problem, Alice has a subset E_A of edges of an undirected graph G on a set V of n vertices, while Bob has a disjoint subset E_B of the edges of $G = (V, E_A, \cup E_B)$, and the players are promised that G is a connected graph. Alice sends a single randomized message $M(E_A)$ to Bob, who should output a spanning tree of G. Note that in the case that G is unweighted, all such spanning trees are minimal.

▶ **Theorem 8.** $R^{1-way}(\mathsf{MST}) = \Omega(n \log n)$.

Proof. We again reduce from Perm on instances of size $n/2$. Alice, given a permutation σ, creates a perfect matching from $[n/2]$ to $[n/2]$ where the i-th left vertex connects to the $\sigma(i)$-th right vertex. Alice's edgeset E_A consists of the edges in this perfect matching. Let L and R denote the two parts of the vertex set V, each of size $n/2$.

Bob's edgeset E_B is just a line connecting the vertices in L. Observe that $G = (V, E_A \cup E_B)$ is connected and has $n-1$ edges, and therefore is itself the only spanning tree of G. Therefore, Bob can reconstruct G. Hence, the players can solve the Perm problem with probability at least $9/10$ given a protocol for MST which succeeds with probability at least $9/10$, and therefore and therefore from Lemma 1, $R^{1-way}(\mathsf{MST}) = \Omega(n \log n)$. ◀

▶ Remark. There is an $O(n \log n)$ bits of space upper bound for MST for integer weights bounded by $\text{poly}(n)$, see section 2.1 of [28].

3.7 k-Edge Connectivity

▶ **Theorem 9.** $D^{1-way}(k\text{-Edge Connectivity}) = \Omega(nk \log n)$.

Proof. Consider a bipartite graph with parts L and R. L is partitioned into $k/2$ blocks L_i each of n/k vertices. Similarly R is partitioned into $k/2$ blocks R_i each of n/k vertices. For each pair (L_i, R_j) containing a left block and a right block, we have a random perfect matching between the blocks. Alice has all of these edges. Bob is interested in the t-th bit of the neighbor of vertex a in the block R_b.

We show a $kn \log n$ lower bound for deterministic protocols. Bob guesses each vertex c in R_b to see if it the neighbor of vertex a in R_b; since the protocol is deterministic it does not err, and so he will figure out the correct neighbor and thus reconstruct the graph as follows. Suppose c is the current candidate. Bob adds edges connecting vertices in the set $(L \setminus a) \cup (R \setminus c)$ to make the graph on these vertices k-edge-connected. This can be done without edge duplications by introducing a clique of k new vertices, and connecting all vertices in $(L \setminus a) \cup (R \setminus c)$ to each of these k new vertices. Bob also adds a set W of $O(k)$ additional vertices and places a k-connected graph H on vertex set $\{a, c\} \cup W$. The resulting graph is k-edge-connected iff there is an edge $\{a, c\}$; if there is such an edge, then by deleting $k/2-1$ neighbors of a and $k/2-1$ neighbors of c, one deletes in total $k-2$ edges and causes H to be disconnected from the rest of the graph. On the other hand if there is no such edge, then at least k edges need to be deleted.

Hence, Bob reconstructs the input graph, which by construction has $\Omega(nk \log n)$ bits of entropy, since there are $(k/2)^2$ random perfect matchings, so the logarithm of the number of possible graphs is $\log_2(((n/k)!)^{k^2/4})$, which gives the desired $\Omega(nk \log n)$ bits of entropy lower bound. This implies $D^{1-way}(k\text{-Edge Connectivity}) = \Omega(nk \log n)$. ◀

▶ Remark. There is a deterministic upper bound of $O(kn \log n)$ bits. See Theorem 1 in [13].

3.8 k-Vertex Connecvitiy

▶ **Theorem 10.** $D^{1-way}(k\text{-Vertex Connectivity}) = \Omega(nk \log n)$.

Proof. Consider a bipartite graph with parts L and R. L is partitioned into $k-1$ blocks L_i each of $n/(k-1)$ vertices. Similarly R is partitioned into $k-1$ blocks R_i each of $n/(k-1)$ vertices. For each pair of left block and right block (L_i, R_j), we have a random perfect matching between the blocks. Alice has all of these edges. Bob is interested in the t-th bit of the neighbor of a in the block R_b. Bob guesses each vertex c in R_b to see if c is the neighbor of a in R_b; since the protocol is deterministic, it does not err, so Bob will figure out the correct neighbor as follows. Suppose c is the current candidate.

Bob adds k new vertices and connects every vertex except a to all k new vertices. Bob also puts a clique on the k new vertices. Finally, Bob adds the edge $\{a, c\}$ to the graph.

Then if $\{a, c\}$ existed in the graph before Bob added it, then vertex a still has only $k-1$ neighbors and so the graph is disconnected by deleting these $k-1$ neighbors. On the other hand, if $\{a, c\}$ did not exist in the graph, then vertex a now has k neighbors and the graph is k-vertex connected.

Thus, since the protocol is deterministic, Bob can reconstruct the input graph, which has $\Omega(kn \log n)$ bits of entropy by construction. This shows $D^{1-way}(k\text{-Vertex Connectivity}) = \Omega(nk \log n)$. ◀

▶ Remark. There is a streaming algorithm due to Eppstein et al. [15] (see also [19] for a discussion) which includes a new edge $\{a, b\}$ iff there are no k-vertex disjoint paths connecting a to b among the edges already stored. Correctness follows from Menger's theorem for vertex connectivity. Note that the algorithm is insensitive to edge duplications, and is deterministic. It achieves $O(kn \log n)$ bits of space.

3.9 H-minor-free

Let H be a fixed graph. In the H-minor-free problem Alice has a subset E_A of edges of an undirected graph G on a set V of n vertices and Bob has a subset E_B of the edges of G. Alice sends a single randomized message $M(E_A)$ to Bob, who should decide if the graph $(V, E_A \cup E_B)$ is H-minor-free. Bob should succeed with probability at least $9/10$.

▶ **Theorem 11.** *For any fixed graph H with minimum degree at least 2, $R^{1-way}(H\text{-minor-free}) = \Omega(n \log n)$.*

Proof. We again reduce from Perm on instances of size $n/2$. Alice, given a permutation σ, creates a perfect matching from $[n/2]$ to $[n/2]$ where the i-th left vertex connects to the $\sigma(i)$-th right vertex. Alice's edgeset E_A consists of the edges in this perfect matching. Let L and R denote the two parts of the vertex set V, each of size $n/2$.

Suppose Bob has the input i to Perm. This corresponds to the ℓ-th bit in $\sigma(j)$ for some $j \in [n/2]$ and $\ell \in [\log(n/2)]$. Bob creates his input to H-minor-free from i and H as follows. Suppose H has $r+1$ vertices $h_0, h_1, \ldots, h_r$. Since $\delta(H) \geq 2$, w.l.o.g., we assume there are two edges $\{h_0, h_1\}$ and $\{h_0, h_2\}$ in $E(H)$. Bob creates r new vertices $p_1, \ldots, p_r$ and puts a copy of $H \setminus \{h_0, h_1\}$ between j and $p_1, \ldots, p_r$ with the mapping $h_0 \to j$ and $h_i \to p_i$ $(i = 1, \ldots, r)$, i.e., j, $p_1, \ldots, p_r$, is an isomorphism to H except for the one edge (j, p_1). Let $V = L \cup R \cup \{p_1, \ldots, p_r\}$ and $S \subset R$ denote the subset of vertices whose ℓ-th bit is equal to 1. Bob also includes all edges in E_B from p_1 to all vertices in S.

Now we claim that there is an H-minor in $(V, E_A \cup E_B)$ iff the ℓ-th bit of $\sigma(j)$ is 1. Indeed, if the ℓ-th bit of $\sigma(j)$ is 1, then there is a edge between $\sigma(j)$ and p_1 in E_B. We can

contract the edges $\{j, \sigma(j)\}$ and $\{\sigma(j), p_1\}$ and we obtain a copy of H. Hence H is a minor of $E_A \cup E_B$.

For the case that the ℓ-th bit of $\sigma(j)$ is 0, then $\sigma(j) \notin S$ and j is not adjacent to any vertex in S. Note that we can delete all isolated matching edges since $\delta(H) \geq 2$. Also since j is not adjacent to a vertex in S and H has minimum degree at least 2, we can contract all edges incident to S, and then contract all nodes in S to p_1. These operations preseve the property of having an H-minor since the minimum degree of H is at least 2. We can also contract $\sigma(j) \in R$ to j since $\deg(\sigma(j)) = 1$ and $\delta(H) \geq 2$. At this point we are left with vertices $p_1, ..., p_r$, and j, with edgeset exactly equal to that of H except we are missing the edge (p_1, j). This implies H is not a minor. ◀

▶ Remark. Kostochka [25] shows that an H-minor-free graph has at most $O(n|H|\sqrt{\log |H|})$ edges. Storing all these edges can be done using $O(n \log n)$ bits. So our lower bound is tight.

As a corollary, we show that the Planar problem in which Bob want to decide if $(V, E_A \cup E_B)$ is planar also has a lower bound of $\Omega(n \log n)$ bits.

▶ **Corollary 12.** $R^{1-way}(\mathsf{Planar}) = \Omega(n \log n)$.

Proof. Consider $H = K_5$ in the previous proof. The graph $(V, E_A \cup E_B)$ is either contracted to a K_5, or a K_5 with one missing edge, according to the ℓ-th bit of $\sigma(j)$. Notice that the former one is non-planar and the latter is planar. Therefore, Planar is as hard as Perm. ◀

▶ Remark. There is an $O(n \log n)$ bit upper bound for Planar, simply store up to $3n$ edges and use that any graph with more than $3n$ edges cannot be planar.

3.10 Approximate Min-Cut

In this section we show an $\Omega(n \log^2 n)$ lower bound for 1-way protocols which provide a constant-factor approximation with constant probability to the minimum cut value of a graph with integer edge weights between 1 and n. Our lower bound also implies an $\Omega(n \log^2 n)$ bit lower bound for $O(1)$-approximate cut sparsifiers of such graphs, as such sparsifiers can be used to approximate the minimum cut value. We let c-approx Min-Cut denote the problem of approximating the minimum cut up to a factor of $c > 1$.

▶ **Theorem 13.** *Suppose a graph has edges with weights in the set $\{1, 2, \ldots, W\}$, where W is at most $2^{\gamma n}$ for a sufficiently small constant $\gamma > 0$. Then for any constant $c > 1$, $R^{1-way}(c\text{-approx Min-Cut}) = \Omega(n \log n \log W)$. In particular, if $W = n$, then $R^{1-way}(c\text{-approx Min-Cut}) = \Omega(n \log^2 n)$.*

Proof. We can reduce the r-AugmentedPerm problem to c-approx Min-Cut, which we abbreviate as the Min-Cut problem in the remainder of the proof. Let $\alpha = 2c + 1$ and $r = \log_\alpha W$. Suppose Alice is given r random permutations $\sigma^1, \ldots, \sigma^r$ of size $n/2$. As in the construction in the proof of Conn, Alice creates r perfect matchings from $\sigma^1, \ldots, \sigma^r$ as her input to the Min-Cut problem. All edge weights in the i-th instance are equal to α^i $(i = 1, \ldots, r)$. The largest weight is $\alpha^r = W$. In expectation there will be $O(r^2)$ duplicate edges when we overlay the matchings. Alice can send the identities of all the duplicate edges together with which instances they occur in to Bob, and not include these in her graph. This only requires $O(\log^2 W(\log n + \log log W))$ additional communication from Alice to Bob, using our choice of r. This is negligible given the upper bound on W in the theorem statement.

Suppose in the r-AugmentedPerm problem Bob is given an index $i \in [n \log n]$, and index $k \in [r]$, and $\sigma^{k+1}, \ldots, \sigma^r$. For the Min-Cut problem, Bob will delete all the edges in the

matchings corresponding to $\sigma^{k+1}, \ldots, \sigma^n$. Bob, depending on which instances he deletes from r-AugmentedPerm, can decide which of the duplicate edges to put back in Alice's graph. As in the Conn problem, Bob also adds a spanning tree to the vertices $(L \setminus \{j\}) \cup S$ in the matching corresponding to σ_k.

Now if $\sigma_i^k = 0$, then the graph is connected. Hence the minimum cut is at least α^k. On the other hand, if $\sigma_i^k = 1$, then the k-th instance is disconnected. In the instances corresponding to $\sigma^1, \ldots, \sigma^{k-1}$, all the vertices have degree one, so if we cut $\{j, \sigma^k(j)\}$ from other vertices, the total weight of this cut is at most $2(\alpha + \alpha^2 + \cdots + \alpha^{k-1}) < \frac{2 \cdot \alpha^k}{\alpha - 1} = \frac{\alpha^k}{c}$. Therefore, if Bob can c-approximate the total weight of the min-cut, then he can distinguish the case $\sigma_i^k = 0$ from $\sigma_i^k = 1$, i.e., he can solve r-AugmentedPerm. ◀

Acknowledgements. We thank Andrew McGregor for helpful discussions regarding this work.

References

1 Kook Jin Ahn, Sudipto Guha, and Andrew McGregor. Graph sketches: sparsification, spanners, and subgraphs. In *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2012, Scottsdale, AZ, USA, May 20-24, 2012*, pages 5–14, 2012.

2 Alexandr Andoni, Huy L. Nguyên, Yury Polyanskiy, and Yihong Wu. Tight lower bound for linear sketches of moments. In *Automata, Languages, and Programming – 40th International Colloquium, ICALP 2013, Riga, Latvia, July 8-12, 2013, Proceedings, Part I*, pages 25–32, 2013.

3 Khanh Do Ba, Piotr Indyk, Eric Price, and David P. Woodruff. Lower bounds for sparse recovery. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pages 1190–1197, 2010.

4 Brian Babcock, Shivnath Babu, Mayur Datar, Rajeev Motwani, and Jennifer Widom. Models and issues in data stream systems. In *Proceedings of the Twenty-first ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 3-5, Madison, Wisconsin, USA*, pages 1–16, 2002.

5 Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. *J. Comput. Syst. Sci.*, 68(4):702–732, 2004.

6 Joshua D. Batson, Daniel A. Spielman, and Nikhil Srivastava. Twice-ramanujan sparsifiers. *SIAM J. Comput.*, 41(6):1704–1721, 2012.

7 Vladimir Braverman, Jonathan Katzman, Charles Seidell, and Gregory Vorsanger. An optimal algorithm for large frequency moments using o(n^(1-2/k)) bits. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2014, September 4-6, 2014, Barcelona, Spain*, pages 531–544, 2014.

8 Amit Chakrabarti, Graham Cormode, and Andrew McGregor. A near-optimal algorithm for estimating the entropy of a stream. *ACM Transactions on Algorithms*, 6(3), 2010.

9 Amit Chakrabarti, Yaoyun Shi, Anthony Wirth, and Andrew Chi-Chih Yao. Informational complexity and the direct sum problem for simultaneous message complexity. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 270–278, 2001.

10 Kenneth L. Clarkson and David P. Woodruff. Numerical linear algebra in the streaming model. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 – June 2, 2009*, pages 205–214, 2009.

11 Peter Clifford and Ioana Cosma. A simple sketching algorithm for entropy estimation over streaming data. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2013, Scottsdale, AZ, USA, April 29 – May 1, 2013*, pages 196–206, 2013.

12 T. Cover and J. Thomas. *Elements of Information Theory*. John Wiley and Sons, Inc., 1991.

13 Michael S. Crouch, Andrew McGregor, and Daniel Stubbs. Dynamic graphs in the sliding-window model. In *Algorithms – ESA 2013 – 21st Annual European Symposium, Sophia Antipolis, France, September 2-4, 2013. Proceedings*, pages 337–348, 2013.

14 T.A. Dowling and R.M. Wilson. Whitney number inequalities for geometric lattices. *Proc. Amer. Math. Soc.*, 47:504–512, 1975.

15 David Eppstein, Zvi Galil, Giuseppe F. Italiano, and Amnon Nissenzweig. Sparsification – a technique for speeding up dynamic graph algorithms. *J. ACM*, 44(5):669–696, 1997.

16 Joan Feigenbaum, Sampath Kannan, Andrew McGregor, Siddharth Suri, and Jian Zhang. On graph problems in a semi-streaming model. In *Automata, Languages and Programming: 31st International Colloquium, ICALP 2004, Turku, Finland, July 12-16, 2004. Proceedings*, pages 531–543, 2004.

17 Joan Feigenbaum, Sampath Kannan, Andrew McGregor, Siddharth Suri, and Jian Zhang. Graph distances in the streaming model: the value of space. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2005, Vancouver, British Columbia, Canada, January 23-25, 2005*, pages 745–754, 2005.

18 Sumit Ganguly. Polynomial estimators for high frequency moments. *CoRR*, abs/1104.4552, 2011.

19 Sudipto Guha, Andrew McGregor, and D. Tench. Vertex and hyperedge connectivity in dynamic graph streams. In *PODS*, 2015.

20 Nicholas J. A. Harvey, Jelani Nelson, and Krzysztof Onak. Sketching and streaming entropy via approximation theory. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 489–498, 2008.

21 T. S. Jayram and David P. Woodruff. Optimal bounds for johnson-lindenstrauss transforms and streaming problems with subconstant error. *ACM Transactions on Algorithms*, 9(3):26, 2013.

22 Daniel M. Kane, Jelani Nelson, and David P. Woodruff. On the exact space complexity of sketching and streaming small norms. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pages 1161–1178, 2010.

23 Daniel M. Kane, Jelani Nelson, and David P. Woodruff. An optimal algorithm for the distinct elements problem. In *Proceedings of the Twenty-Ninth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2010, June 6-11, 2010, Indianapolis, Indiana, USA*, pages 41–52, 2010.

24 Michael Kapralov, Yin Tat Lee, Cameron Musco, Christopher Musco, and Aaron Sidford. Single pass spectral sparsification in dynamic streams. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 561–570, 2014.

25 Alexandr Kostochka. The minimum hadwiger number for graphs with a given mean degree of vertices. *Metody Diskret. Analiz.*, 38:37–58, 1982.

26 Ilan Kremer, Noam Nisan, and Dana Ron. On randomized one-round communication complexity. In *Proceedings of the Twenty-Seventh Annual ACM Symposium on Theory of Computing, 29 May-1 June 1995, Las Vegas, Nevada, USA*, pages 596–605, 1995.

27 Yi Li and David P. Woodruff. A tight lower bound for high frequency moment estimation with small error. In *Approximation, Randomization, and Combinatorial Optimization.*

Algorithms and Techniques – 16th International Workshop, APPROX 2013, and 17th International Workshop, RANDOM 2013, Berkeley, CA, USA, August 21-23, 2013. Proceedings, pages 623–638, 2013.

28 Andrew McGregor. Graph stream algorithms: a survey. *SIGMOD Record*, 43(1):9–20, 2014.

29 S. Muthukrishnan. Data streams: Algorithms and applications. *Foundations and Trends in Theoretical Computer Science*, 1(2), 2005.

30 Eric Price and David P. Woodruff. (1 + eps)-approximate sparse recovery. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 295–304, 2011.

31 Ran Raz and Boris Spieker. On the "log rank"-conjecture in communication complexity. In *34th Annual Symposium on Foundations of Computer Science, Palo Alto, California, USA, 3-5 November 1993*, pages 168–176, 1993.

A Information Theory Facts

For a discrete random variable X with possible values $\{x_1, x_2, \ldots, x_n\}$, the Shannon entropy of X is defined as $H(X) = -\sum_{i=1}^{n} \Pr(X = x_i) \log_2 \Pr(X = x_i)$. Let $H_b(p) = -p \log_2 p - (1-p) \log_2 (1-p)$ denote the binary entropy function when $p \in (0, 1)$. For two random variables X and Y with possible values $\{x_1, x_2, \ldots, x_n\}$ and $\{y_1, y_2, \ldots, y_m\}$, respectively, the conditional entropy of X given Y is defined as $H(X \mid Y) = \sum_{i,j} \Pr(X = x_i, Y = y_j) \log_2 \frac{\Pr(Y=y_j)}{\Pr(X=x_i, Y=y_j)}$. Let $I(X;Y) = H(X) - H(X \mid Y) = H(Y) - H(Y \mid X)$ denote the mutual information between two random variables X, Y. Let $I(X;Y \mid Z)$ denote the mutual information between two random variables X, Y conditioned on Z, i.e., $I(X;Y \mid Z) = H(X \mid Z) - H(X \mid Y, Z)$.

The following summarizes several basic properties of entropy and mutual information.

▶ **Proposition 14.** *Let X, Y, Z, W be random variables.*

1. *If X takes value in $\{1, 2, \ldots, m\}$, then $H(X) \in [0, \log m]$.*
2. *$H(X) \geq H(X \mid Y)$ and $I(X;Y) = H(X) - H(X \mid Y) \geq 0$.*
3. *If X and Z are independent, then we have $I(X;Y \mid Z) \geq I(X;Y)$. Similarly, if X, Z are independent given W, then $I(X;Y \mid Z, W) \geq I(X;Y \mid W)$.*
4. *(Chain rule of mutual information) $I(X, Y; Z) = I(X;Z) + I(Y;Z \mid X)$.*
   More generally, for any random variables $X_1, X_2, \ldots, X_n, Y$,
   $I(X_1, \ldots, X_n; Y) = \sum_{i=1}^{n} I(X_i; Y \mid X_1, \ldots, X_{i-1})$.
   Thus, $I(X, Y; Z \mid W) \geq I(X; Z \mid W)$.
5. *(Fano's inequality) Let X be a random variable chosen from domain $\mathcal{X}$ according to distribution μ_X, and Y be a random variable chosen from domain $\mathcal{Y}$ according to distribution μ_Y. For any reconstruction function $g : \mathcal{Y} \to \mathcal{X}$ with error δ_g,*

$$H_b(\delta_g) + \delta_g \log(|\mathcal{X}| - 1) \geq H(X \mid Y).$$