CMSC 858M: Fun with Hardness Proofs
Spring 2021
Chpt. 20 Lower Bounds on Streaming Algorithms

Instructor: Mohammad T. Hajiaghayi, William Gasarch
Scribe: Jacob Gilbert

April 2021

1 Overview

This chapter focuses on achieving lower bounds mainly using communication complexity for streaming algorihtms, dynamic algorithms with limited memory but varying number of passes. Note again here that a chapter link is broken in the overview when referencing Chapter 19.

2 Introduction to Streaming Algorithm

The title of section 20.2 is missing an r in the word streaming. Besides this, the explanation of streaming algs is clear.

3 Streaming for Graph Algorithms

Again this section is just definitions and is clear.

4 Streaming Algorithm for Maximal Matching

The proof of a 2-approx streaming algorithm for Maximal Matching is simple and therefore easily understandable.

5 Communication Complexity

Definitions of the three communication complexity problems are understandable. In step 2 of Definition 20.5.2, I think "number of bits needed" should say "number of bits needed for communication" to clarify its just communication we care about.

6 Lower Bounds on Graph Streaming Problems

In general the four proofs in this section are very understandable as they include graphs for each complexity proof. I believe the graph of the Perfect-Matching proof, Thm 20.6.4, has the edges for Bob and Alice switched. Also, I may suggest swapping the order of Perfect-Matching and Shortest-Path only because they have similar proofs but Shortest-Path provides the entire proof while Perfect-Matching asks the reader to figure out the argument in step 5. Thus, having Shortest Path first would be a nice warmup to Perfect Matching potentially.

7 Additional 10 Complexity Problems

From Mayur Datar, Aristides, Gionis, Piotr Indyk, and Rajeev Motwani, "Maintaining Stream Statistics over Sliding Windows":

1. **Basic Counting** - given a stream of bits, maintain a count of the number of 1's in the last N elements seen from the stream. This problem requires $\Omega(\frac{1}{\epsilon}\log^2 N)$ memory for any deterministic/randomized algorithms.

2. **Sum** - given a stream of integers in range $[R]$, maintain the sum of the last N integers. This problem requires $\Omega(\frac{1}{\epsilon}(\log^N + \log R \log N))$ bits for any streaming algorithm. Note that this and the previous problem relate to computing the L_p norm with an underlying vector that has a single dimension.

From Rajesh Chitnis, Graham Cormode, MohammadTaghi Hajiaghayi, and Morteza Monemizadeh, "Parameterized Streaming Algorithms for Vertex Cover":

Parameterized Vertex Cover - Is there a vertex cover of a graph in the streaming setting with at most size k? There is a lower bound of $\Omega(k^2)$ space required for any randomized streaming algorithm for parameterized vertex cover, even when limited to edge insertions.

From Elad Verbin, Wei Yu "The Streaming Complexity of Cycle Counting, Sorting By Reversals, and Other Problems":

1. **Boolean Hidden Matching (BHM)** - Alice is given an n-bit string $x \in \{0,1\}^n$ and Bob gets a perfect matching M on n vertices. Thus, the n bits of Alice are matched up in pairs but only Bob knows the matching. The goal is to determine which of two cases holds: either all the matched-up pairs of bits XOR to 1, or XOR to 0. Verbin and Yu use a generalized version of this problem, Boolean Hidden Hypermatching to prove many streaming results by showing a communication complexity of $\Omega(n^{1-1/t})$ where the matching is instead a t-uniform hypermatching.

2. **Cycle counting** - Alice gets a perfect matching E_A on a bipartite graph with n vertices on each side and Bob gets a perfect matching E_B on the same graph. The union $E_A \cup E_B$ is a collection of disjoint cycles. The goal of the problem is to approximate the number of cycles and decide if it is $\leq a$ or $\geq b$. Proven by a reduction from Boolean Hidden Hypermatching, the communication complexity is $\sqrt{n}$.

3. **Sorting by reversal on signed permutations** - Given a data stream of a permutation S on $\{1, ..., n\}$, a reversal $r(i, j)$ will transfer $x = (x_1, ..., x_n)$ to $(x_1, ..., x_{i-1}, -x_j, ..., -x_i, x_{j+1}, ...x_n)$. Find the minimum number of reversals needed to sort S. This problem requires space $\Omega((n/8)^{1-1/t})$ for approximation factor $1 + 1/(4t - 2)$.

From Sepehr Assadi, Vishvajeet N, "Graph Streaming Lower Bounds for Parameter Estimation and Property Testing via a Streaming XOR Lemma:

1. **Minimum Spanning Tree estimation** - Given a weighted undirected graph, estimate the weight of the minimum spanning tree in G. $\Omega(1/\epsilon)$ passes are needed for $n^{o(1)}$ space for even constant weights.

2. **ϵ-Connectivity** - If at least $\epsilon \cdot n$ edges need to be inserted into G to make it connected, G is said to be ϵ-far from being connected. There needs at least $n^{o(1)}$ space in $\Omega(1/\epsilon)$ passes to solve solve this in the streaming setting.

3. **Cycle-freeness** - If at least $\epsilon \cdot n$ edges need to be deleted from G to remove all its cycles, then G is said to be ϵ-far from being cycle-free. Similarly, there needs at least $n^{o(1)}$ space in $\Omega(1/\epsilon)$ passes for any streaming algorithm.

4. **Pointer Chasing (PC)** - $PC_{m,b}$ has a (m, b)-layered graph on layers $V_1, ..., V_{b+1}$, arbitrary vertex $s \in V_1$, and an arbitrary equipartition X, Y of V_{b+1}. The goal is to decide whether the set of vertices in V_{b+1} reachable by s belongs to X or Y. If a PC streaming algorithm succeeds with probability at least $\frac{1}{2} + \gamma$ for some $\gamma \in (0, 1/2)$, then there are either at least b passes or there is at least $\Omega(\frac{\gamma^4}{b^5} \cdot m)$ memory.