COMPSCI 630: Randomized Algorithms February 21, 2018

Lecture 11

Lecturer: Rong Ge *Scribe: Hanrui Zhang*

1 Overview

In this lecture, we introduce an application of random projection — the *second moment* problem in the *streaming* model, and sketch a randomized scheme using Johnson-Lindenstrauss. We then study *communication complexity* and discuss its relationship with streaming. In particular, we prove upper/lower bounds for the deterministic/randomized communication complexity of the function of equality of two boolean strings, and discuss the implication of a deterministic lower bound on the second moment problem.

2 Introduction

Recall the statement of Johnson-Lindenstrauss:

> Any n vectors can be compressed into an $O(\log n/\varepsilon^2)$-dimensional space with distances/inner products preserved.

It appears natural that Johnson-Lindenstrauss can be used to save time. We consider here its power in saving space, and an application of this power.

3 The Streaming Model and the Second Moment Problem

Certain computational tasks involve massive data arriving online (e.g. a router keeping statistics of the traffic). The streaming model, as an abstraction of such tasks, allows a single scan through the input data, and a limited amount of space compared to the length of the input.

One archetypal problem in the streaming model is the second moment problem:

Definition 1 (The Second Moment Problem (Informal))**.** *Given a vector $x^{(0)} = \mathbf{0}$ of m dimensions, a sequence of t updates of form (i_k, Δ_k) arrive online. After the k-th update, the current vector $x^{(k-1)}$ becomes*

$$x^{(k)} = x^{(k-1)} + \Delta_k e_{i_k}$$

where e_j is the vector whose j-th coordinate is 1 and other coordinates are 0. The problem asks to approximately compute $\|x^{(k)}\|_2^2$ after each update.

Consider a solution via random projection: Let A be a random matrix in $\mathbb{R}^{d\times m}$ ($d = O(\log t/\varepsilon^2)$), where $A_{ij} \sim N(0,1)$ (or ± 1.) By Johnson-Lindenstrauss for $y^{(k)} = \frac{1}{\sqrt{d}} A x^{(k)}$, w.h.p. $\|y^{(k)}\| \approx \|x^{(k)}\|$. So maintaing $y^{(k)}$ of dimension d, and updating by

$$y^{(k)} = y^{(k-1)} + \frac{\Delta_k}{\sqrt{d}} A e_{i_k},$$

"solve" the problem — if we can store A.

The remedy for this is to make the columns of A p-wise independent instead of being i.i.d., which is possible with $O(pd \log m)$ space by storing seeds for the entries. In particular, $p = 4$ already suffices for a some weaker guarantee.

4 Communication Complexity

Consider the following scenario: Alice has a number $x \in \{0,1\}^n$, and Bob has a number $y \in \{0,1\}^n$. They want to compute $f(x,y)$ depending on both x and y by sending bits to each other. Roughly speaking the bits required to be sent measures how hard f is to compute in terms of communication.

Definition 2 (Deterministic Communication Complexity (Informal))**.** *The deterministic communication complexity of a function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ is defined to be*

$$D(f) := \min_{\text{ALG for } f} \max_{x,y \in \{0,1\}^n} \#\text{ of bits sent},$$

where $\text{ALG}(x,y) = f(x,y)$ *for all x, y.*

To get a sense of communication complexity consider the function of equality:

$$\text{EQ}(x,y) = \begin{cases} 1, & \text{if } x = y \\ 0, & \text{otherwise} \end{cases}.$$

Clearly $D(\text{EQ}) \leq n$ because Alice can always send x to Bob and Bob can then check if $x = y$. In fact, for any f, $D(f) \leq n$. We show:

Proposition 1. $D(\text{EQ}) = n$.

Proof. Fix an arbitrary ALG which (1) always outputs the correct value and (2) sends $t < n$ bits. Let $T(x,y)$ be the "transcript" sent by ALG on input (x,y). Consider the transcripts on inputs $E = \{(i,i) \mid i \in \{0,1\}^n\}$, (abusing notations) denoted by $T(E)$. Note two facts:

- ALG always outputs 1 on any input in E.
- $T(E) \subseteq \{0,1\}^t$, so $|T(E)| < 2^n$ and there are i, j where $i \neq j$ and $T(i,i) = T(j,j)$.

Now consider running ALG on input (i,j). It is easy to see that $T(i,j) = T(i,i) = T(j,j)$, so deterministic Alice and Bob cannot tell the difference from (i,i) or (j,j) and consequently output 1, which is incorrect, leading to a contradiction. □

We now consider randomized protocols, which are significantly more powerful (at least for EQ).

Definition 3 (Randomized Communication Complexity (Informal))**.** *The randomized communication complexity of a function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ is defined to be*

$$R(f) := \min_{\text{ALG for } f} \max_{x,y \in \{0,1\}^n} \#\text{ of bits sent},$$

where $\Pr[\text{ALG}(x,y) = f(x,y)] \geq \frac{2}{3}$ *for all x, y.*

We further assume that Alice and Bob have access to public coins, i.e. a same public random string, for two reasons: (1) it simplifies the analysis and (2) the private coin setting can be reduced to the public coin setting without too much loss.

Consider the same function EQ. We show:

Proposition 2. $R(\text{EQ}) \leq 2$.

Proof. Consider this simple protocol (which we call ALG):

1. Alice observes two random strings u, $v \in \{0,1\}^n$, and sends $\langle u,x\rangle$ and $\langle v,x\rangle$ to Bob;
2. Bob observes the same strings u and v, and computes $\langle u,y\rangle$ and $\langle v,y\rangle$. Bob outputs 1 iff $\langle u,x\rangle = \langle u,y\rangle$ and $\langle v,x\rangle = \langle v,y\rangle$.

When $x = y$ the protocol never fails. When $x \neq y$ let $i \in [n]$ be such that $x_i \neq y_i$.

$$\Pr[\langle u,x\rangle = \langle u,y\rangle] = \Pr[\langle u_i, x_i - y_i\rangle = \langle u_{-i}, x_{-i} - y_{-i}\rangle] = \Pr[u_i = \langle u_{-i}, x_{-i} - y_{-i}\rangle] = \frac{1}{2}.$$

Since u and v are independent,

$$\Pr[\mathrm{ALG}(x,y) = 0] = 1 - \Pr[\langle u,x\rangle = \langle u,y\rangle]^2 = \frac{3}{4} \geq \frac{2}{3}.$$

□

5 Relationship between Streaming and Communication Complexity

Consider a stronger version of the streaming model, which is also a weaker version of the communication model: Alice reads the first half of the input and sends her memory to Bob, and Bob reads the second half and generates the output. This is stronger than the streaming model because within the first/second half, an unlimited amount of memory is allowed. This is weaker than the communication model because only one round of communication is allowed. We leverage this model to show:

Proposition 3. *The second moment problem with a t-dimensional vector and $2t$ updates requires at least t bits of memory, even if the indices of the updates are consecutive (i.e. $i_k = (k-1) \bmod n+1$) and the differences are ± 1. Note that the restricted form of input requires no more than $2t$ bits.*

Proof. Suppose there is an algorithm using at most $t-1$ bits. First we relax the problem to the two-halves version: the algorithm reads the first t updates, erases its memory except the first $t-1$ bits, reads the rest t updates, and outputs $\|x\|_2^2$. This means if Alice is given the first t updates and Bob is given the last t, then there is a protocol which computes $\|x\|_2^2$ by sending no more than $t-1$ bits.

Consider the restricted form of input. It is easy to see that $\|x\|_2^2 = 0$ iff $\Delta_k + \Delta_{k+t} = 0$ for all $k \in [t]$. This protocol can be used to compute EQ: Alice maps each bit of her input to $\{-1,1\}$ (for example, 0 to 1 and 1 to -1), runs the protocol and sends her memory to Bob. Now Bob maps his input to $\{-1,1\}$ correspondingly (i.e. 0 to -1 and 1 to 1), and runs the protocol on the mapped input and Alice's memory. Finally Bob outputs 1 iff $\|x\|_2^2 = 0$.

It is easy to check this protocol for EQ is deterministic, correct, and sends at most $t-1$ bits, which contradicts $D(\mathrm{EQ}) = t$. □