**CS 15-851: Algorithms for Big Data** Spring 2025

# Lecture 10 — 3/27/2024

*Prof. David Woodruff* *Scribe: Aksara Bayyapu*

## Randomized 1-Way Communication Complexity

In this section, we study the *one-way communication complexity* for the Index problem. We consider the following scenario:

- Alice holds a binary string $x \in \{0,1\}^n$.
- Alice sends a single message $M$ to Bob.
- Bob has an index $j \in [n]$ and must determine $x_j$.

The protocol is *randomized*, meaning Alice and Bob may use randomness when constructing and interpreting $M$. We require that for every $x$ and every $j$, Bob outputs the correct value of $x_j$ with probability at least $\frac{2}{3}$ over the randomness of the protocol:

$$\forall x, \forall j, \quad \mathop{\mathbb{P}}_{\text{randomness}}[\text{Bob correctly identifies } x_j] \;\geq\; \tfrac{2}{3}.$$

Our goal is to prove that any protocol (i.e., any strategy for choosing and interpreting $M$) that achieves this success criterion must send at least $\Omega(n)$ bits of information from Alice to Bob.

## 1-Way Communication Complexity of Index

We consider the Index problem in a one-way communication setting:

- Let $X \in \{0,1\}^n$ be chosen uniformly at random.
- Alice observes $X$ and sends a single message $M$ (of $\ell$ bits) to Bob.
- Bob has an index $j \in [n]$ and must predict the bit $X_j$.

We require that for all $j$, Bob's prediction $X'_j$ satisfies

$$\mathbb{P}[X'_j = X_j] \;\geq\; \tfrac{2}{3},$$

where the probability is over the randomness of $X$ (and any randomness in the protocol).

To understand how much information $M$ reveals about $X_j$, we use *Fano's Inequality*. Since $X \to M \to X'$ is a Markov chain (i.e., $X_j$ and $X'_j$ become conditionally independent given $M$), Fano's Inequality implies:

$$H(X_j \mid M) \;\leq\; H(P_e) + P_e \log_2(|X_j| - 1),$$

where $P_e = \mathbb{P}[X'_j \neq X_j]$ is the error probability. Given $\mathbb{P}[X'_j = X_j] \geq \frac{2}{3}$, we have $P_e \leq \frac{1}{3}$. In the binary case ($|X_j| = 2$), this simplifies to

$$H(X_j \mid M) \;\leq\; H\Big(\tfrac{1}{3}\Big) \;+\; \tfrac{1}{3}\,\log_2(2-1) \;=\; H\Big(\tfrac{1}{3}\Big).$$

Hence each bit $X_j$ is not completely hidden once $M$ is known; $M$ must reveal some information about $X_j$.

We now lower-bound the *mutual information* $I(X; M)$. Since $X$ is $n$ bits chosen uniformly and the bits are independent:

$$I(X; M) \;=\; \sum_{i=1}^{n} I(X_i; M \mid X_{<i}) \;=\; n - \sum_{i=1}^{n} H(X_i \mid M,\, X_{<i}).$$

Because conditioning on more variables can only reduce entropy,

$$H(X_i \mid M,\, X_{<i}) \;\leq\; H(X_i \mid M),$$

we get

$$I(X; M) \;\geq\; n - \sum_{i=1}^{n} H(X_i \mid M) \;\geq\; n - n\,H\Big(\tfrac{1}{3}\Big) \;=\; \Omega(n).$$

Since $M$ is an $\ell$-bit message, we know $H(M) \leq \ell$. Moreover, by the data-processing inequality,

$$I(X; M) \;\leq\; H(M).$$

Thus,

$$\ell \;\geq\; H(M) \;\geq\; I(X; M) \;=\; \Omega(n).$$

Therefore, any one-way protocol that correctly identifies $X_j$ with probability at least $2/3$ for all $j$ must use messages of length at least $\Omega(n)$ bits.

## Typical Communication Reduction

A common strategy for proving lower bounds in computational problems is to reduce the problem in question to a known problem with established complexity bounds, such as the *Index* problem. The following figure illustrates this framework in the context of a streaming algorithm.

1. Alice runs a streaming algorithm on $s(a)$: She processes her stream $s(a)$ using some algorithm Alg, then sends the *state* of $\text{Alg}(s(a))$ to Bob.

2. Bob computes $\text{Alg}(s(a), s(b))$: After receiving the state from Alice, Bob processes his own stream $s(b)$ together with the state to produce an output.

3. Space complexity versus communication complexity: If Bob successfully solves a function $g(a, b)$ with this setup, it implies that the space used by Alg must be at least as large as the one-way communication complexity of $g$. In other words, a lower bound on the one-way communication complexity of $g$ transfers to a lower bound on the space complexity of the streaming algorithm.

## Example: Distinct Elements

We have a sequence of elements $a_1, a_2, \ldots, a_m \in [n]$ and wish to compute the total number of *distinct* elements in this sequence. We claim that any algorithm (or protocol) achieving this requires $\Omega(n)$ bits of communication in the one-way setting.

Recall the *Index* problem:

- Alice holds a binary string $x \in \{0,1\}^n$.
- Bob holds an index $i \in [n]$.
- Bob's goal is to determine whether $x_i = 1$.

We reduce the Distinct Elements problem to Index as follows:

1. Let $s(a)$ be the stream of indices $\{i_j\}$ such that $x_{i_j} = 1$ in $x$. That is, $s(a)$ only lists positions where $x$ has a 1-bit.
2. Let $s(b) = i$, the single index held by Bob.
3. Suppose we have a streaming algorithm Alg that computes the number of distinct elements. We feed $s(a)$ into Alg, producing a (possibly) compressed state $\text{Alg}(s(a))$.
4. Then we feed $s(b)$ to Alg as well, obtaining $\text{Alg}(s(a), s(b))$.

Define

$$x_i \;=\; \begin{cases} 0, & \text{if } \text{Alg}(s(a), s(b)) \;=\; \text{Alg}(s(a)) + 1, \\ 1, & \text{otherwise.} \end{cases}$$

- If $x_i = 1$, then the index $i$ was already included in $s(a)$, so when we add $s(b) = i$, the total count of distinct elements *does not* increase. Hence, $\text{Alg}(s(a), s(b)) = \text{Alg}(s(a))$.
- If $x_i = 0$, then $i$ was not included in $s(a)$, so adding $i$ to the stream increases the count of distinct elements by 1. Hence, $\text{Alg}(s(a), s(b)) = \text{Alg}(s(a)) + 1$.

By correctly detecting whether $\text{Alg}(s(a), s(b))$ differs from $\text{Alg}(s(a))$, Bob effectively learns $x_i$. This shows that a protocol solving the Distinct Elements problem in one pass (with a one-way message) can solve the Index problem. Consequently, the space (or communication) complexity of Alg is at least the one-way communication complexity of Index, namely $\Omega(n)$ bits.

## Strengthening Index: Augmented Indexing

In the *Augmented Index* problem:

- Alice holds a binary string $x \in \{0,1\}^n$.
- Bob has an index $i \in [n]$ **and** the first $i-1$ bits of $x$, namely $x_1, x_2, \ldots, x_{i-1}$.

- Bob's goal remains the same as in the standard Index problem: to determine $x_i$ with probability at least $1-\delta$ (or some constant success probability $> \frac{1}{2}$).

Even with the extra information $x_1, \ldots, x_{i-1}$ available to Bob, the one-way communication complexity of the Augmented Index problem remains $\Omega(n)$. Concretely, one can show

$$\mathrm{CC}_\delta(\text{Augmented Index}) \;\geq\; n\big(1 - H(\delta)\big),$$

where $H(\delta) = -\delta \log_2(\delta) - (1-\delta)\log_2(1-\delta)$ is the binary entropy function.

Let $M$ be the (randomized) message Alice sends to Bob. We use the chain rule for mutual information:

$$I(X; M) \;=\; \sum_{i=1}^{n} I(X_i; M \mid X_{<i}).$$

Since $X$ is uniform in $\{0,1\}^n$ and the bits are independent,

$$I(X_i; M \mid X_{<i}) \;=\; H(X_i \mid X_{<i}) \;-\; H(X_i \mid M, X_{<i}) \;=\; 1 - H\big(X_i \mid M, X_{<i}\big).$$

Hence,

$$I(X; M) \;=\; \sum_{i=1}^{n} \Big[1 - H\big(X_i \mid M, X_{<i}\big)\Big] \;=\; n - \sum_{i=1}^{n} H\big(X_i \mid M, X_{<i}\big).$$

Bob's task is to predict $X_i$ given $M$ and $X_{<i}$. If Bob can guess $X_i$ correctly with probability at least $1-\delta$, then by Fano's Inequality,

$$H\big(X_i \mid M, X_{<i}\big) \;\leq\; H(\delta) + \delta \log_2\big(|\mathcal{X}_i| - 1\big).$$

Since $X_i$ is a single bit ($|\mathcal{X}_i| = 2$), this simplifies to

$$H\big(X_i \mid M, X_{<i}\big) \;\leq\; H(\delta).$$

Combining the above,

$$\sum_{i=1}^{n} H\big(X_i \mid M, X_{<i}\big) \;\leq\; \sum_{i=1}^{n} H(\delta) \;=\; n\, H(\delta).$$

Thus,

$$I(X; M) \;\geq\; n - n\, H(\delta) \;=\; n\big(1 - H(\delta)\big).$$

Since $I(X; M) \leq H(M) \leq |M|$ (the length of the message in bits), we conclude

$$|M| \;\geq\; n\big(1 - H(\delta)\big).$$

Hence,

$$\mathrm{CC}_\delta(\text{Augmented Index}) \;\geq\; n\big(1 - H(\delta)\big),$$

establishing that even with partial knowledge of $x$, Bob still needs an $\Omega(n)$-bit message from Alice to reliably determine the unknown bit $x_i$.

## Log n Bit Lower Bound for Estimating Norms

We consider a scenario where Alice has input $x \in \{0,1\}^{\log(n)}$ in the Augmented Index framework. The goal is to show that estimating a norm (e.g., $\|\cdot\|_p$ for $p \geq 1$) within a factor of 2 requires at least $\Omega(\log n)$ bits of communication, despite the fact that naively storing the counter (or norm) might take only $O(\log(\log(n)))$ bits.

Alice constructs a vector $v \in \mathbb{R}^d$ (for some dimension $d$) that has a *single* non-zero coordinate. Specifically, if $x = (x_1, x_2, \ldots, x_{\log(n)})$, she sets

$$v = \Big( \sum_{j=1}^{\log(n)} 10^j x_j,\ 0,\ 0,\ \ldots,\ 0 \Big).$$

Alice processes $v$ with a streaming (or data-structure) algorithm Alg and sends the resulting state $\mathrm{Alg}(v)$ to Bob.

Bob has:

- An index $i \in [\log(n)]$.
- The subsequent bits $x_{i+1}, x_{i+2}, \ldots, x_{\log(n)}$ (as in an Augmented Index setting).
- He constructs a second vector $w$ whose single non-zero coordinate is $\sum_{j>i} 10^j$.

Bob then *subtracts* $w$ from $v$ by feeding $-w$ into the algorithm's state:

$$\mathrm{Alg}(v - w) \ = \ \mathrm{Alg}\Big( \sum_{j \leq i} 10^j x_j \Big).$$

The outcome of $\mathrm{Alg}(v-w)$ gives Bob enough information to guess $x_i$:

$$x_i \ = \ \begin{cases} 1, & \text{if } \mathrm{Alg}(v-w) \ \geq \ \frac{10^i}{2}, \\ 0, & \text{otherwise.} \end{cases}$$

If $x_i = 1$, then $v - w = \sum_{j \leq i} 10^j x_j$ is at least $10^i$, so any 2-approximation to $\|v-w\|_p$ must be $\geq \frac{10^i}{2}$. If $x_i = 0$, then $v - w$ is at most $\sum_{j<i} 10^j < 2 \cdot 10^{i-1}$, so a 2-approximation stays below $\frac{10^i}{2}$. Hence, Bob can distinguish between $x_i = 0$ and $x_i = 1$ via the approximate norm.

## $\frac{1}{\varepsilon^2}$ Bit Lower Bound for Norm Estimation

The *Gap Hamming Distance* problem is defined over two strings $x, y \in \{0,1\}^n$. We are guaranteed that

$$\Delta(x,y) \ = \ \sum_{k=1}^{n} \mathbf{1}\{x_k \neq y_k\}$$

is either $> \frac{n}{2} + 2\varepsilon n$ or $< \frac{n}{2} + \varepsilon n$, and the task is to decide which of these two cases holds.

It is known that solving the Gap Hamming problem in a one-way randomized communication model requires $\Omega\big(\frac{1}{\varepsilon^2}\big)$ bits of communication. Moreover, one can show that approximating certain *norms*

(e.g. $\|\cdot\|_p$) on binary data streams reduces to the Gap Hamming problem, implying a similar lower bound for norm estimation.

We illustrate how the Gap Hamming problem can be related to *Index* by a public-coin argument:

- Public Coins: Suppose we have a $t \times t$ random matrix $r_j^k$ (for rows $k = 1, \ldots, t$ and columns $j = 1, \ldots, t$). We view each row $r^k = (r_1^k, \ldots, r_t^k)$ as a random bit vector.
- Alice's Input: A bitstream $x \in \{0,1\}^t$ (with $t = O(\frac{1}{\varepsilon^2})$).
- Bob's Input: An index $i \in \{1, \ldots, t\}$ (or a set of bits $b \in \{0,1\}^t$ derived from $i$).

We construct two new bit vectors $a, b \in \{0,1\}^t$ as follows:

$$a_k \;=\; \text{Majority}\big\{\, r_j^k : x_j = 1\big\}, \quad b_k \;=\; r_i^k.$$

If $x_i = 0$, the $i$-th column $r_i^k$ does *not* affect $a_k$ (since $x_i = 0$ means the $i$-th column is not included in the majority vote for $a_k$). Thus, $a$ and $b$ are essentially *uncorrelated* along the $i$-th dimension.s If $x_i = 1$, then the $i$-th column *does* contribute to $a_k$, creating correlation between $a$ and $b$.

Denote $\Delta(a,b) = \sum_{k=1}^t \mathbf{1}\{a_k \neq b_k\}$.

Case $x_i = 0$: The $i$-th column is not part of $a$'s majority computation. For each $k$, $a_k$ equals the majority of other columns where $x_j = 1$, and $b_k = r_i^k$ is an independent random bit. One can show that in expectation, half of these bits disagree, giving

$$\mathbb{E}\big[\Delta(a,b)\big] \;\approx\; \tfrac{t}{2}.$$

Case $x_i = 1$: Now the $i$-th column is included in $a$'s majority vote. Analyzing the probability that $b_k = r_i^k$ agrees with the majority bit of $\{r_j^k : x_j = 1\}$ yields a difference on the order of $\Theta(\varepsilon t)$ from the $\frac{t}{2}$ baseline. Concretely,

$$\mathbb{E}\big[\Delta(a,b)\big] \;\approx\; \tfrac{t}{2} - \Theta(\varepsilon t) \;=\; \tfrac{t}{2} - \Theta\Big(\tfrac{1}{\varepsilon}\Big).$$

Since $t = \Theta(\frac{1}{\varepsilon^2})$, these two cases yield a noticeable gap in $\Delta(a,b)$ on the order of $\sqrt{t} \approx \frac{1}{\varepsilon}$. This gap is sufficient to distinguish whether $x_i = 0$ or $x_i = 1$ by checking if $\Delta(a,b)$ is closer to $\frac{t}{2}$ or $\frac{t}{2} - \Theta(\sqrt{t})$.

Because distinguishing $x_i = 0$ from $x_i = 1$ (an Index-like task) can be embedded in deciding which side of the gap $\Delta(a,b)$ falls on (a Gap Hamming problem), we see that solving this problem requires $\Omega(\frac{1}{\varepsilon^2})$ bits of communication. Therefore, the same applies for the norm estimation problem.