Streaming Algorithms

Meng-Tsung Tsai

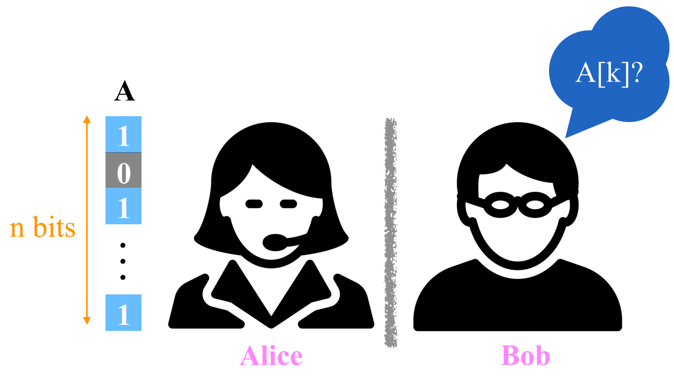
05/22/2018

References

- "Communication Complexity," Kushilevitz
- "Lower Bounds for One-Way Communication," Roughgarden

Randomized Communication Complexity

The indexing problem



How many bits are needed to be sent from Alice to Bob so that Bob can figure out whether A[k] = 1 or not w.p. > 7/8 for every single input? (Ω(n) bits)

Distributional Complexity

Let P be a distribution over $\{0, 1\}^n \times \{0, 1\}^n$; that is, P describes the probability that input = (x, y) for each $(x, y) \in \{0, 1\}^n \times \{0, 1\}^n$.
Let f be the function to be computed.

If every deterministic one-way protocol D with

$$\Pr_{(x, y)\sim P}[D(x, y) \neq f(x, y)] \leq \varepsilon$$

requires at least k bits, then every (public coin) randomized one-way protocol R with (two-sided) error $\leq \varepsilon$ requires at least k bits.

Distributional Complexity

Proof.

Let R be a randomized protocol, i.e. a distribution over deterministic protocols $D_1(R)$, $D_2(R)$, ..., $D_s(R)$.

If R requires < k bits to answer with error $\leq \varepsilon$, then each deterministic protocol $D_i(R)$ uses < k bits. By the assumption, each $D_i(R)$ answers with error $> \varepsilon$.

Then R answers with error $> \varepsilon$.

Deterministic Protocol that Allows Errors

Claim. If a deterministic protocol D for Index problem sends at most cn bits (c is a sufficiently small constant and n is sufficiently large) and the input is sampled uniformly, then the probability that D incurs an error is at least 1/8.

Proof. For each message **z** that Alice sends to Bob, depending only on **z** and k, Bob has to answer A[k] = 0 or 1. Let $b(z) \in \{0, 1\}^n$ be Bob's answer when the transmitted message is **z**.

Let S(**z**) be the set of Alice's input so that the transmitted message is **z**.

- At most one element in S(**z**) has 0 disagreement with b(**z**).
- At most n elements in S(**z**) have only 1 disagreement with b(**z**).
- At most C(n, k) elements in S(**z**) have exactly k disagreement with b(**z**).

Deterministic Protocol that Allows Errors

Proof. There are 2^{cn} distinct messages **z**, and # elements in S(**z**) for all **z** that have an error w.p. $\leq n/4$ is at most

$$2^{cn}\left(\binom{n}{0}+\binom{n}{1}+\cdots+\binom{n}{n/4}\right) \leq n2^{cn}(4e)^{n/4} \leq 2^{(c+0.87)n+\log n}$$

Pick c < 0.03 and n > 100 so that ((c+0.87)n+log n) < n-1.

Hence, at least 2^{n-1} (half) distinct inputs incur an error w.p. $\geq 1/4$.

For sufficiently large n, any D that uses 0.03 n bits has an error rate at least 1/8.

$\Rightarrow$ For sufficiently large n, any R that has an error rate < 1/8 for every single input requires $\Omega(n)$ bits. QED

Allowing Higher Error Rates

If R(Index) = o(n) when the randomized protocol R incurs an error w.p. $< 1/2 - \varepsilon$ for an arbitrary small constant $\varepsilon > 0$, then we run a constant copies of R independently in parallel still requires o(n) bits.

On the other hand, the error rate drops to $< 1/8$. $\rightarrow\leftarrow$

$\boxed{R(\text{Index}) = \Omega(n).}$

Space Lower Bounds for Randomized Streaming Algorithms

Topological Sort

Input: a directed acyclic graph D

Output: a node ordering $v_1, v_2, \ldots, v_n$ so that for every arc (u, v) in D, node u appears earlier than v in the ordering.

Goal: show that any 1-pass randomized streaming algorithm that can output the ordering w.p. $> 1/2+\varepsilon$ for any constant $\varepsilon > 0$ requires $\Omega(n^2)$ bits.

<u>Remark</u>. The naive algorithm that stores the entire graph in an adjacentcy matrix turns out to be optimal.

Reduction

Conduct a reduction from Index(C(n, 2)) to T-sort.

Construct a graph with 2n nodes, 1x, 2x, ..., nx, 1y, 2y, ..., ny, initially without any edge.

Look at the problem instance of the Index problem. If the (i*n+j)-th bit is 0, then add an edge from ix to jy, or otherwise add an edge from jx to iy.

Reduction

To figure whether the (i*n+j)-th bit is 0 or 1, we add two edges (iy, ix) and (jy, jx).

Claim. The resulting graph is still acyclic. (Why?)

If node iy appears before node jx, then the (i*n+j)-th bit is 0; otherwise iy appears after node jx, then the (i*n+j)-th bit is 1.

Sorting

Input: n integers.

Output: the input integers in the sorted order.

Approach: Let Alice's input be an array of length 2n and let half of them be 1-bits. In such cases, R(Sorting) still has a lower bound $\Omega(n)$. Then we reduce Index to Sorting.

$$\boxed{R(\text{Sorting}) = \Omega(n).}$$

Closest Pair

Input: n points on a plane.

Output: the pair of points whose distance is shortest.

$$\boxed{R(\text{ClosestPair}) = \Omega(n).}$$