6.5240 Sublinear Time Algorithms	October 19, 2022
Lecture 11	
Lecturer: Ronitt A. Rubinfeld	*Scribe: Ethan Zahid*

1 Overview

This lecture looked at two main things. First it defined Yao's principle, which tells us that an average case deterministic lower bound on query complexity is a randomized worst case lower bound on query complexity. Next we considered the problem of determining whether a string is the concatenation of two palindromes and lower bounded the query complexity of any algorithm that solves this problem as $\Omega(\sqrt{n})$.

2 Yao's Principle

Theorem 1 *Suppose there exists a distribution $\mathcal{D}$ on pass/fail inputs such that any deterministic decider with $\leq t$ query complexity is wrong with probability $p \geq \frac{1}{3}$ on input uniformly randomly chosen from $\mathcal{D}$. Then t is a lower bound on the complexity of a randomized decider for the same query.*

Proof Consider a problem over inputs $\mathcal{X}$ and $\mathcal{A}$ be the set of all possible deterministic algorithms that solve the problem. For $a \in \mathcal{A}, x \in \mathcal{X}$ let $c(a, x)$ be the cost of running algorithm a on input x. Then Yao's principle claims that for $A \in \mathcal{A}, X \in \mathcal{X}$ chosen from some distributions on $\mathcal{A}, \mathcal{X}$

$$\max_{x \in \mathcal{X}} E[c(A, x)] \geq \min_{a \in \mathcal{A}} E[c(a, X)]$$

which is just a special case of von Neumann's minimax theorem.

■

3 Palindrome Concatenation

3.1 Problem

Note that the problem of determining whether or not a string x is a palindrome is pretty simple. We repeatedly sample i from $[n]$ and check that $x_i = x_{n+1-i}$ and reject if this is ever not the case. By making $O(\frac{1}{\epsilon})$ we can get a very good algorithm since for a string that is ϵ-far from being a palindrome, each sample has probability at least ϵ of causing the algorithm to reject and so after $\frac{1}{\epsilon}$ samples you expect the algorithm to reject. What about determining whether or not a string x is the concatenation of two palindromes?

Let $L_n = \{w | w \in \{0,1\}^n, w = vv^R uu^R\}$ be the set of strings that are the concatenation of two palindromes. Define w to be ϵ-close to L_n if $\exists\, w' \in L_n$ such that w, w' differ in $\leq \epsilon n$ places.

Theorem 2 *An algorithm A must make $\Omega(\sqrt{n})$ queries if it satisfies that*

$$\forall x \in L_n \;\; \Pr[A(x) = \text{Pass}] \geq \frac{2}{3}$$

$$\forall x \; \epsilon\text{-far from } L_n \;\; \Pr[A(x) = \text{Fail}] \geq \frac{2}{3}$$

The rest of the notes will be dedicated to proving the above theorem.

3.2 Distributions

First we define three distributions as follows

Distribution N

- Output uniformly randomly from all strings ϵ-far from L_n

Distribution P

- Pick $k \in [\frac{n}{6} + 1, \frac{n}{3}]$
- Generate random v, u such that $|v| = k, |u| = \frac{n}{2} - k$
- Output $vv^R uu^R$

Distribution $\mathcal{D}$

- Output from N with probability $\frac{1}{2}$ and from P with probability $\frac{1}{2}$

3.3 Error

Any deterministic algorithm A works by making successive queries and decides what query to make next based on the result of the previous queries. Using t queries there are 2^t sequences of queries/results we can make (assuming binary results to queries), call these 2^t queries the root-leaf paths of A. Each of these 2^t leaves will output pass or fail according to A.

Now for a leaf l we define the following two errors of l

- $E^-(l) = \{\text{inputs } w \text{ } \epsilon\text{-far from } L_n \text{ that reaches } l\}$
- $E^+(l) = \{\text{inputs } w \in L_n \text{ that reaches } l\}$

$$\text{Total Error on } \mathcal{D} = \sum_{\text{pass } l} \Pr[w \in E^-(l)] + \sum_{\text{fail } l} \Pr[w \in E^+(l)]$$

Claim 3 *If $t = o(n)$, $\forall\ l$ at depth t*

$$\Pr_D[w \in E^-(l)] \geq \left(\frac{1}{2} - o(1)\right) \cdot 2^{-t}$$

Proof Since there are $2^{n/2}$ choices for u, v and $\frac{n}{2}$ choices for k

$$|L_n| \leq 2^{n/2} \cdot \frac{n}{2}$$

Let P_n be the set of w that are ϵ-close to L_n, if for each element of L_n we consider all strings we can get by changing it in r places for $r \in [\epsilon n]$ we get all elements of P_n, therefore

$$|P_n| \leq 2^{n/2} \cdot \frac{n}{2} \cdot \sum_{r=0}^{\epsilon n} \binom{n}{r} \leq 2^{n/2} \cdot \frac{n}{2} \cdot \epsilon n \cdot \binom{n}{\epsilon n}$$

Since $\binom{n}{r}$ for $r \in [\epsilon n]$ is maximized at $r = \epsilon n$. Next see that an application of Stirling's Approximation gives the bound

$$\binom{n}{k} \leq \left(\frac{en}{k}\right)^k$$

$$2^{n/2} \cdot \frac{n}{2} \cdot \epsilon n \cdot \binom{n}{\epsilon n} \leq 2^{n/2} \cdot \frac{\epsilon n^2}{2} \cdot \left(\frac{1}{\epsilon}\right)^{\epsilon n} \cdot e^{\epsilon n}$$

$$\begin{aligned} 2^{n/2} \cdot \frac{\epsilon n^2}{2} \cdot \left(\frac{1}{\epsilon}\right)^{\epsilon n} \cdot e^{\epsilon n} &\leq 2^{n/2} \cdot 2^{\log \frac{\epsilon n^2}{2}} \cdot 2^{\log (\frac{1}{\epsilon})^{\epsilon n}} \cdot 2^{\log e^{\epsilon n}} \\ &\leq 2^{n/2} \cdot 2^{\log \epsilon + 2 \log n} \cdot 2^{\epsilon n \log \frac{1}{\epsilon}} \cdot 2^{2\epsilon n} \\ &\leq 2^{n/2 + 2\epsilon n \log \frac{1}{\epsilon}} \end{aligned}$$

Next see that of the 2^n inputs in $\{0,1\}^n$ exactly 2^{n-t} reaches any specific leaf l. This is because each of the t decisions in our query tree splits the input space in half until 2^{-t} of the 2^n inputs reach l. This means that

$$|E^-(l)| \geq 2^{n-t} - |P_n| \geq 2^{n-t} - 2^{n/2 + 2\epsilon n \log \frac{1}{\epsilon}} = (1 - o(1)) \cdot 2^{n-t}$$

$$\Pr_D[w \in E^-(l)] = \frac{1}{2} \cdot \frac{|E^-(l)|}{|N|} \geq \frac{1}{2} \cdot \frac{(1 - o(1)) \cdot 2^{n-t}}{2^n} = \left(\frac{1}{2} - o(1)\right) \cdot 2^{-t}$$

where the $\frac{1}{2}$ comes from the probability we choose from N. ■

Claim 4 *If $t = o(\sqrt{n})$, $\forall\ l$ at depth t*

$$\Pr_D[w \in E^+(l)] \geq \left(\frac{1}{2} - o(1)\right) \cdot 2^{-t}$$

Proof In our algorithm, for any input we make t queries. If we consider all $\binom{t}{2}$ pairs of these 2 queries, for each pair of queries there is clearly at most 2 values of k for which the two queries are symmetric around either k or $\frac{n}{2}+k$. Then the number of k such that no two of the t queries are symmetric around k or $\frac{n}{2}+k$ is $\geq \frac{n}{6}-2\binom{t}{2} = \frac{n}{6}(1-o(1))$ for $t = o(\sqrt{n})$. In addition, for each of these values of k there are $2^{n/2}$ inputs from L_n which are evenly split up amongst the 2^t leaves (since the queries are not symmetric around k, $\frac{n}{2}+k$) and so each leaf l has $|E^+(l)| = \frac{n}{6}(1-o(1)) \cdot 2^{\frac{n}{2}-t}$. Now see that

$$\mathrm{Pr}_D[w \in E^+(l)] = \sum_w \sum_k \mathrm{Pr}_D[w|k] \cdot \mathrm{Pr}[\text{choose } k] \cdot \mathbf{1}_{w \in E^+(l)}$$

$$\mathrm{Pr}_D[w|k] = \frac{1}{2} Pr_P[w|k] = \frac{1}{2} \cdot 2^{-n/2}, \ \ \mathrm{Pr}[\text{choose } k] = \frac{1}{\frac{n}{6}} = \frac{6}{n}$$

$$\mathrm{Pr}_D[w \in E^+(l)] = \sum_w \sum_k \frac{1}{2} \cdot 2^{-n/2} \cdot \frac{6}{n} \cdot \mathbf{1}_{w \in E^+(l)} = \frac{1}{2} \cdot \frac{(1-o(1)) \cdot \frac{n}{6} \cdot 2^{n/2-t}}{\frac{n}{6} \cdot 2^{n/2}} = \left(\frac{1}{2} - o(1)\right) \cdot 2^{-t}$$

■

3.4 Conclusion

If $t = o(\sqrt{n})$ then the total error satisfies

$$\text{Total Error on } \mathcal{D} = \sum_{\text{pass } l} \mathrm{Pr}[w \in E^-(l)] + \sum_{\text{fail } l} \mathrm{Pr}[w \in E^+(l)]$$

$$\text{Total Error on } \mathcal{D} \geq \sum_{\text{pass } l} \left(\frac{1}{2} - o(1)\right) \cdot 2^{-t} + \sum_{\text{fail } l} \left(\frac{1}{2} - o(1)\right) \cdot 2^{-t}$$

$$\text{Total Error on } \mathcal{D} \geq \sum_{l} \left(\frac{1}{2} - o(1)\right) \cdot 2^{-t}$$

$$\text{Total Error on } \mathcal{D} \geq \left(\frac{1}{2} - o(1)\right) \gg \frac{1}{3}$$

And so by Yao's Principle any algorithm A that passes $w \in L_n$ with probability $\geq \frac{2}{3}$ and fails w ϵ-far from Ł$_n$ with probability $\geq \frac{2}{3}$ must use $\Omega(\sqrt{n})$ queries, proving the theorem.