Complexity & Algorithms, Spring 2026

More lower bounds

Next topics:

Intro algorithms: Recursion, Dynamic Programming, and Greedy algorithms	11 respondents	**61 %**
Intro to graph algorithms: BFS/DFS, MST, and Shortest paths	11 respondents	61 %
Minimum spanning trees and shortest paths in graphs	9 respondents	50 %
Flows and cuts on graphs, plus applications	3 respondents	17 %
Linear Programming (algorithms, duality, and applications)	6 respondents	33 %
Hashing/streaming/sketching data	7 respondents	39 %
Intro to randomized algorithms	9 respondents	50 %
Advanced data structures (Union-find, Fibonacci heaps, splay trees)	5 respondents	28 %
Approximation algorithms	10 respondents	56 %
Fast Fourier Transforms	7 respondents	39 %
Computational Geometry algorithms	5 respondents	28 %
Lower bounds for problems - adversarial analysis	8 respondents	44 %
Dimensionality reduction - JL Lemma and SVDs	6 respondents	33 %

Gaps in problem:

Sorting: $T(n) \geq \lceil \log(n!) \rceil$

for $n = 2$, $T(n) = 29$

<u>But</u>: merge sort uses 30 comparisons

Can we get better lower bound?

A new problem : Keifer 1953

A function $f : \{0, 1, \ldots, n-1\} \to \mathbb{R}$ is unimodal if:

- $\exists$ a unique maximum x^*
- no two successive values are equal

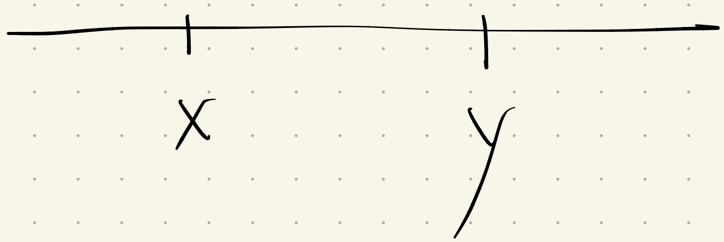
In other words:

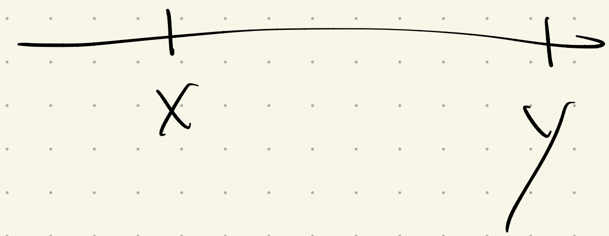
$f(0)$ $\quad\quad$ $f(x^*)$ $\quad\quad$ $f(n-1)$

Goal:

Suppose we know $f(x)$ & $f(y)$:

3 possibilities:





Turn this into decision tree:

Pinning interval $[i, j]$:
an interval containing x^*

Initially: $[0, n-1]$, & our decision tree should shrink it.

Ideas for an algorithm?

Make this precise: $T(n) \le 2\lceil \log_2 n \rceil$

1. Query powers of 2:

Stop when $f(2^k) < f(2^{k-1})$

$\Longrightarrow$

How many queries?

(2) Now, have a pinning interval $[2^{k-1}, 2^k]$

Do binary search:

Query middle, shrink interval by $1/2$

So # queries:

Lower bound: $T(n) \geq \lceil \log_2 n \rceil$

How many leaves in decision tree?

How to get <u>exact</u> answer?

Aside: "Little Birdie Principle"

Here:

To get a tighter bound:
invert the problem

Let $N(t)$ = size of largest pinning range where we can find x^* in $\leq t$ probes

Small t:

interval size:

$t=1$

$t=2$

Our previous bounds:

$$T(n) \leq 2\lceil \log_2 n \rceil$$

$$\Rightarrow \quad N(t) \geq$$

$$T(n) \geq \lceil \log_2 n \rceil$$

$$\Rightarrow \quad N(t) \leq$$

Theorem: $N(t) = F_{t+1} - 1$

where $F_k = k^{th}$ Fibonacci number.

Proof: Induction on t:

Base cases:

IH:

Algorithm ($\geq$):

Let $n = N(t-1) + N(t-2) + 1$, & show can find with t steps!

Probe at 2 spots:

n

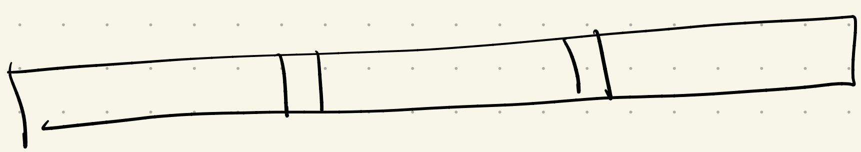
Lower bound:

Suppose you probe at i & j

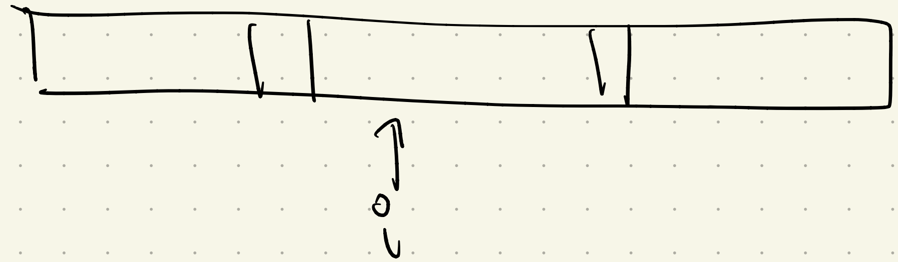
if $i = N(t-2)$ & $j = N(t-1)$

if $i < N(t-2)$:

adversary says "$f(i) < f(j)$"



if $i > N(t-2)$:



say "$f(i) > f(j)$"

and also shares "$x^* < i$"

↳ can only help!

Then