Episode

Sorting Lower Bound

Sorting Algorithms

- $\Theta(n^2)$
  - Selection Sort
  - Bubble Sort
  - Insertion Sort
- $\Theta(n \log n)$
  - Quick Sort
  - Merge Sort
  - Balance Tree
- Guide Question
  - **Can we do better?**

Spaghetti Sort

$O(n)$

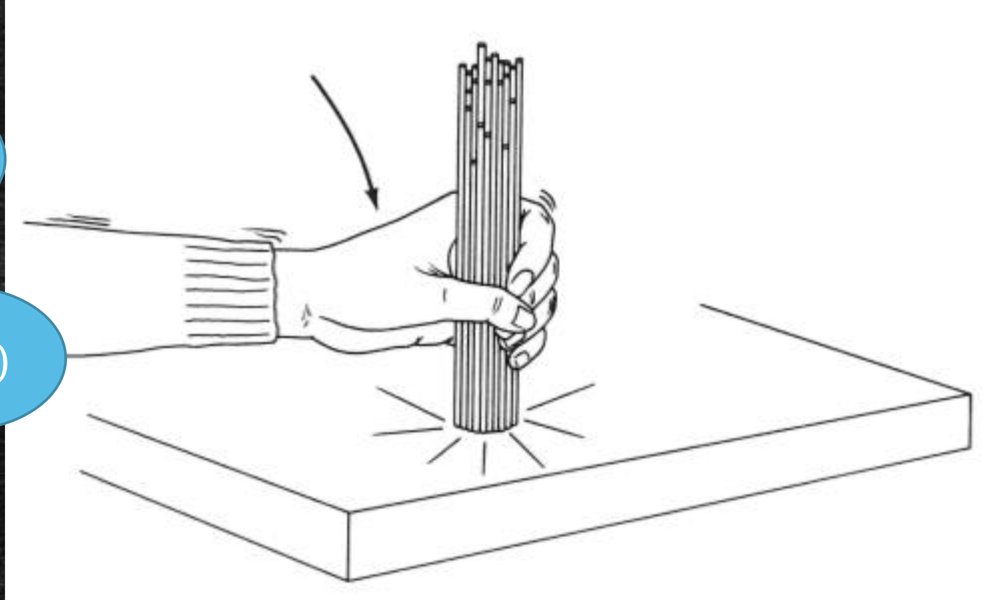
- For each number, **break** a piece of spaghetti whose length is that number.

$O(1)$

- Take all spaghetti and **push** them against the table!

$O(n)$

- Keep **touching** and **removing** spaghetti from the top by your other hand.



Is it what we want?

- Need to think
  - What can we do?
  - What can not we do?

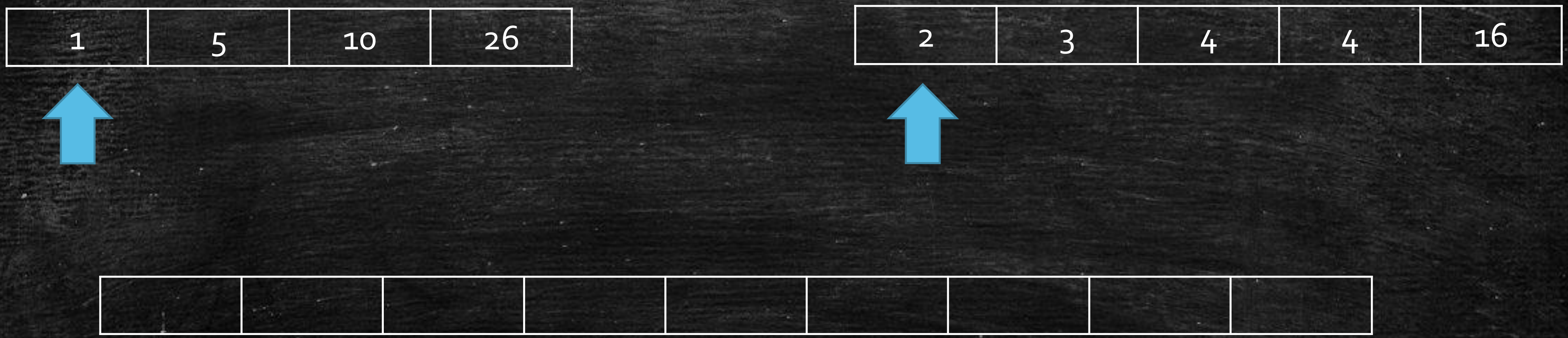
Is it what we want?

- Need to think
  - ~~What can we do?~~
  - ~~What can not we do?~~
  - What can computer do?
  - What can not computer do?
- A proper Computation Model!
  - Allowed Operations: Comparison
  - Not allowed Operations: Break spaghetti, push spaghetti, touch spaghetti.

Comparison-based Sorting

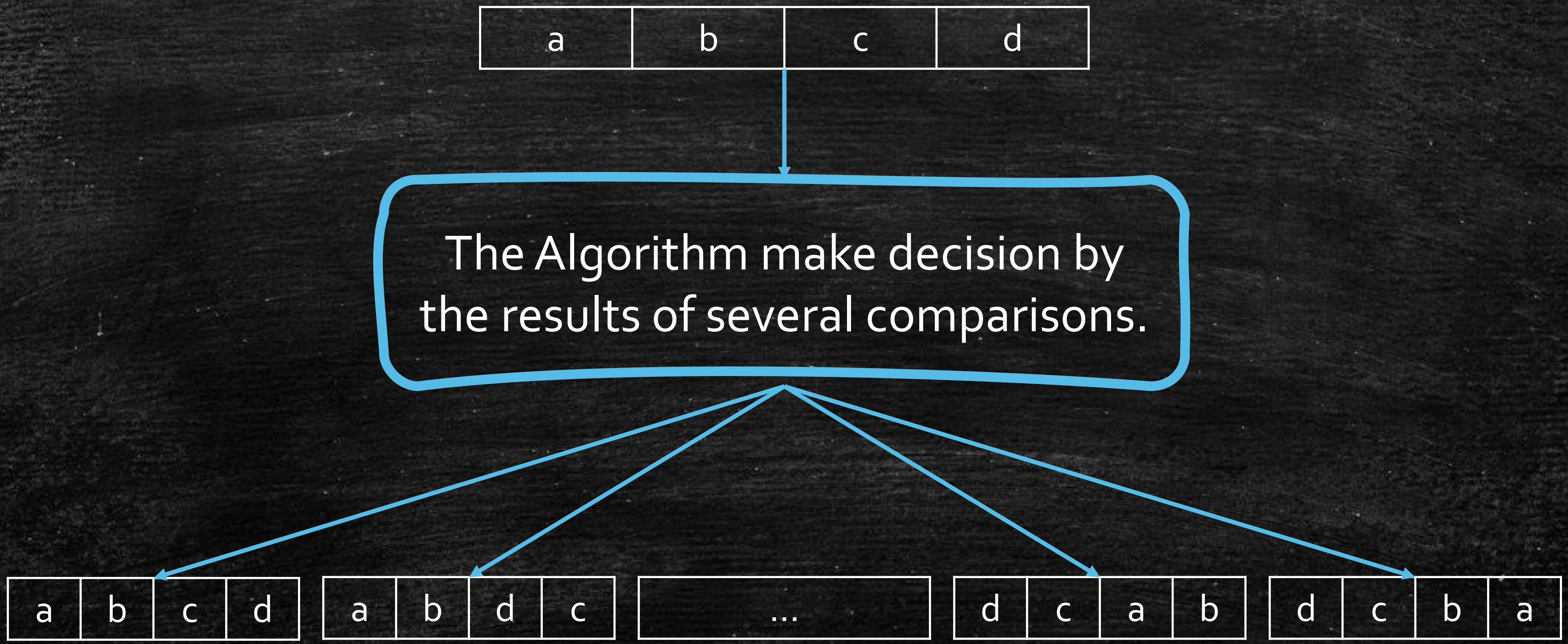
- Only allowed operation: **comparison**.
- Can do
  - $Compare(a, b)$, answer $a > b$ or $b \geq a$.
- Can not do
  - Ask what a is, what b is.
- Examples
  - Merge Sort
  - Insertion Sort
  - Quick Sort
  - ……

Recall Merge Sort



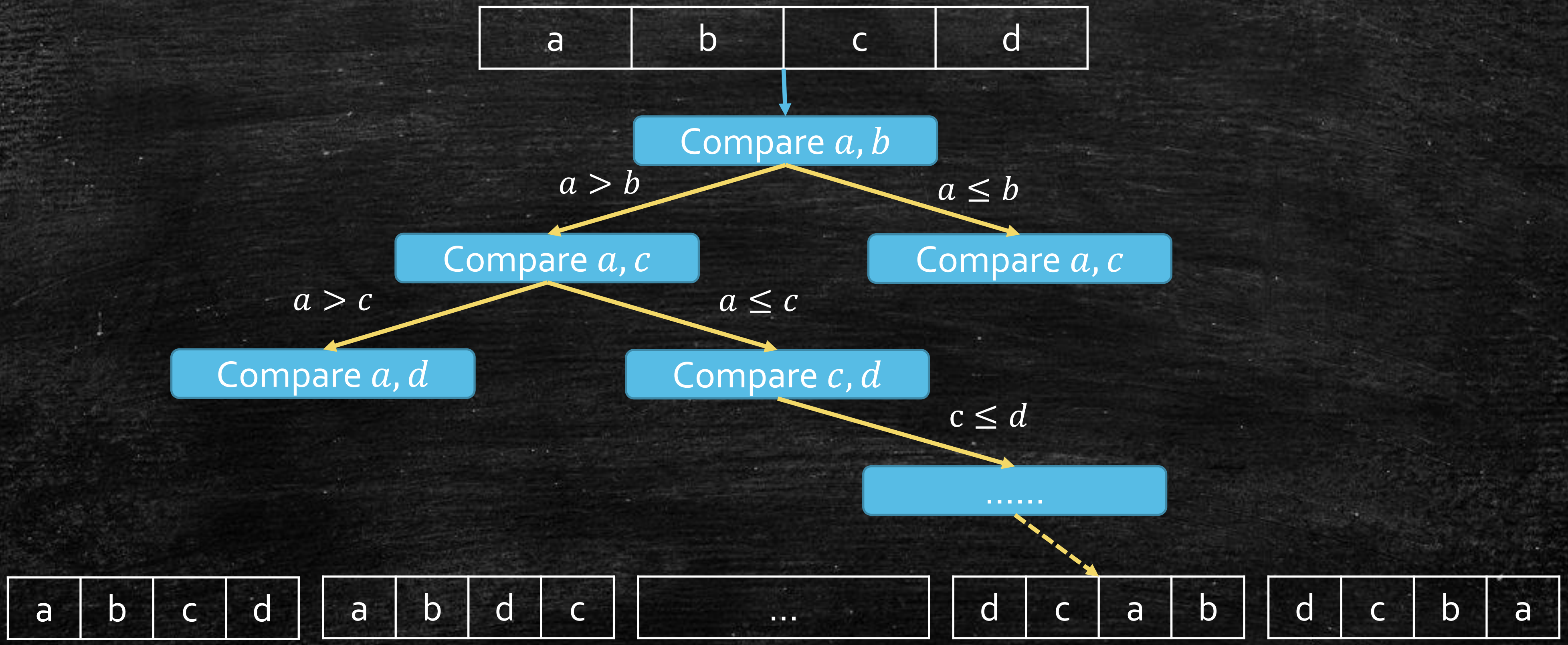
- Plan
  - Maintain 2 pointers $i = 1, j = 1$
  - Repeat
    - Append $\min\{a_i, b_j\}$ to C
    - If a_i is smaller, then move i to $i + 1$; If b_j is smaller, then move j to $j + 1$.
    - Break if $i > n$ or $j > m$
  - Append the reminder of the non-empty list to C

Comparison-based Sorting



- How many possible ordering?

Comparison-based Sorting



- How many possible ordering? --- $n!$

Comparison-based Sorting: Time Complexity

1. Comparison-based Sorting forms a binary tree.
2. It should have at least $n!$ leaves.
3. What is the running time?
4. Worst case: the **longest** path from root to leaves.
5. Best deterministic algorithm make the tree shallowest.
6. Shallowest tree is a balanced tree.
7. Height: $\log(n!)$

Last step!

- The lemma we have
  - Any deterministic comparison-based algorithms must take $\boldsymbol{\log(n!)}$ steps to sort an array in the worst case.
- The theorem we want
  - Any deterministic algorithm comparison-based algorithms must take $\boldsymbol{\Omega(n \log n)}$ time.

Proof

$$\begin{aligned}
\log(n!) &= \log 1 + \log 2 + \log 3 + \cdots + \log n \\
&\geq \log\left(\frac{n}{2} + 1\right) + \log\left(\frac{n}{2} + 2\right) + \cdots + \log n \\
&\geq \frac{n}{2} \log \frac{n}{2} \\
&= \Omega(n \log n)
\end{aligned}$$

Good News!

Merge Sort is the Optimal Deterministic Comparison-based Algorithm

More Questions

- What about **randomized** algorithms
  - Still $\Omega(n \log n)$
- Do we have **linear time** sorting algorithms that is not comparison-based?
  - Yes! But with some restriction.