Sorting

Fall 2020

School of Software Engineering

South China University of Technology

Lower Bound for Sorting

A General Lower Bound for Sorting

- The problem of sorting is bounded by $\Omega(N)$ and $O(N \log N)$.
  - A sorting problem cannot be solved by any algorithm in less than $\Omega(N)$ time
  - Current best known sorting algorithm is in $O(N\log N)$ (for average and worst cases)

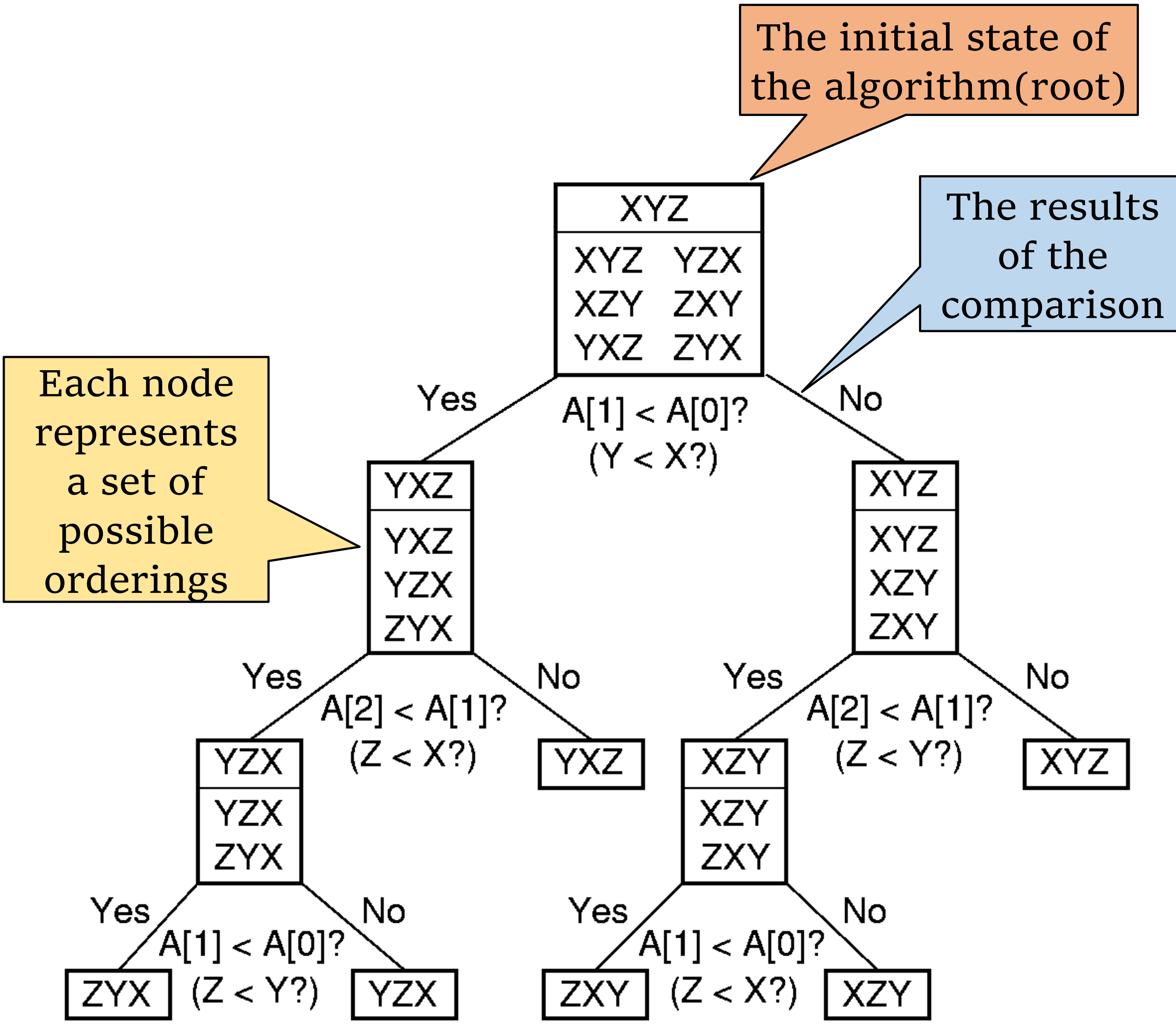
- One of the most important and most useful proofs in computer science

No sorting algorithm based on **key comparisons** can possibly be faster than $\Omega(N \log N)$ in the worst case.

Decision Tree

- Assume that all N elements are distinct
- Any sorting algorithm based on comparisons can be modeled by a **decision tree**.
- A decision tree is a binary tree that can model the processing for any algorithm that makes binary decisions.
  - Binary decision: yes or no; smaller than or larger than
  - Each decision is represented by a branch in the tree

Decision-Tree



Decision tree for Insertion Sort of array A=[X, Y, Z]

Decision-Tree

- All sorting algorithms can be viewed as algorithms to "find" the correct permutation of the input that yields a sorting list.
  - Proceed by making branches in the tree based on the results of key comparisons
  - The algorithm can terminate once a node with a single permutation has been reached (leaf).

- The depth of the deepest node represents the longest series of decisions required by the algorithm to reach an answer
  - It corresponds to the worst-case cost of the algorithm!

Binary tree properties

- A binary tree of depth d has at most 2^d leaves.

- A binary tree with L leaves must have depth at least $\lceil \log L \rceil$.

A General Lower Bound for Sorting

- What is the smallest depth possible for the deepest node in the tree for any sorting algorithm using only comparisons?

- Any sorting algorithm that uses only comparisons requires at least $\lceil \log(N!) \rceil$ comparisons in the worst case.
  - A decision tree to sort N elements must have N! leaves.(Each leaf corresponds to a permutation of N elements)

A General Lower Bound for Sorting

- Any sorting algorithm that uses only comparisons between elements requires Ω (N logN) comparisons.

$$
\begin{aligned}
\log(N!) &= \log(N(N-1)(N-2)\cdots(2)(1)) \\
&= \log N + \log(N-1) + \cdots + \log 2 + \log 1 \\
&\geq \log N + \log(N-1) \cdots + \log(N/2) \\
&\geq (N/2)\log(N/2) \\
&\geq (N/2)\log N - (N/2) \\
&= \Omega(N \log N)
\end{aligned}
$$

Decision-Tree Lower Bounds for Selection Problems

- Selection Problems
  - find the smallest item in an N-element collection
  - find the two smallest items in an N-element collection
  - find the median

- The lower bounds of these problems solved by comparison-based algorithms can be determined using decision trees.

Decision-Tree Lower Bounds for Selection Problems

- Assume all items are unique
- N-k + $\left\lceil \log\binom{N}{k-1} \right\rceil$ comparisons are necessary to find the kth smallest items?

- Any comparison-based algorithm to find the smallest element must use at least $N-1$ comparisons.

Decision-Tree Lower Bounds for Selection Problems

- If all the leaves in a decision tree are at depth d or higher, the decision tree must have at least 2^d leaves.
  - A decision tree is a binary tree
  - A binary tree with depth d must have at most 2^d leaves.
  - Here, all non-leaf nodes have two children.

- The decision tree for finding the smallest of N elements must have at least 2^{N-1} leaves.
  - All leaves in this decision tree are at depth $N - 1$ or higher

Decision-Tree Lower Bounds for Selection Problems

- The decision tree T for finding the kth smallest of N elements must have at least $\binom{N}{k-1}2^{N-k}$ leaves.
  - If t is the kth smallest element
  - $S = \{x_1, x_2, \ldots x_{k-1} \mid x_i < t, i=1,2,\ldots,k-1\}$
  - $R = \{x_k, x_{k+1}, \ldots x_N \mid x_j <= t, j=k,k+2,\ldots,N\}$
  - T' is a decision tree in which does not include the comparisons between any element in S and that in R
  - T' must have at least $2^{|R|-1} = 2^{N-k}$ leaves which are correspond to the elements in S.
  - there are $\binom{N}{k-1}$ choices for S
  - there must be at least $\binom{N}{k-1}2^{N-k}$ leaves in T
- T must have depth at least $\left\lceil \binom{N}{k-1}2^{N-k} \right\rceil$

Homework 5-4

- Self study 7.10 Adversary augment for lower bounds