Comparison Lower Bounds

CS 4231, Fall 2020

Mihalis Yannakakis

Comparison Sorts

- Many sorting algorithms that we saw (Quicksort, Mergesort, Insertion Sort) use at least $\Omega(n \log n)$ time.
- Is this the best possible?

- All these algorithms are comparison sorts:

  only operations on elements are comparisons

  - Algorithms apply to all ordered domains, No special assumptions on the domain of the elements

Lower bound for Comparison Sorts

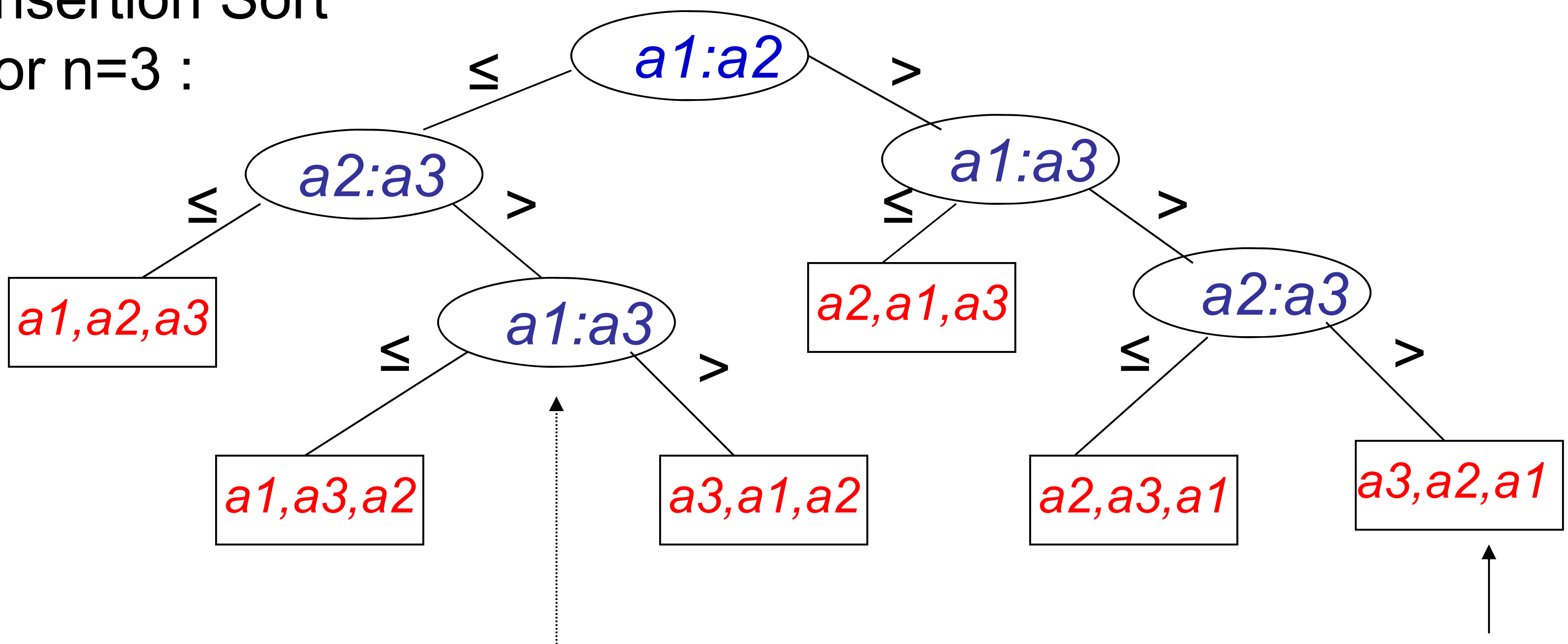
Theorem: Every comparison sort must make Ω(n logn) comparisons in the worst case.

Same lower bound applies to

- average case for uniformly random input permutations
- expected time of randomized algorithms

Decision (Comparison) Tree

Insertion Sort
for n=3 :



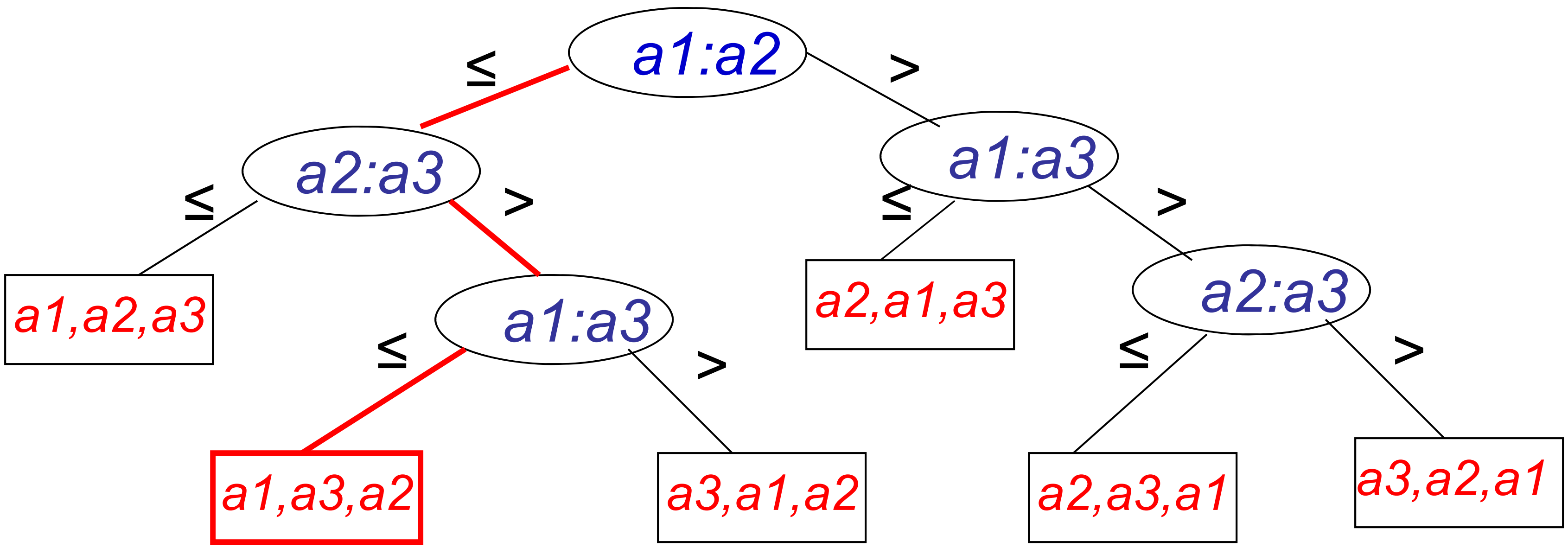
comparison: $a_1 \leq a_3$?

answer: $a_3 \leq a_2 \leq a_1$

Comparison-based sorting algorithm $\rightarrow$

A decision (comparison) tree for every n

Input → Path to a leaf



Example: Input $[a_1, a_2, a_3] = [2, 7, 5]$

Answer: $a_1 \le a_3 \le a_2$

Lower bound

- # comparisons for an input = length of path
- Worst-case complexity (in #comparisons) = height of the tree (assuming no useless leaves)
- A leaf for each permutation $\Rightarrow$ n! leaves
- Every binary tree with L leaves has height ≥ logL
- Height of the decision tree ≥ log(n!)
- Stirling's formula: $n! = \sqrt{2\pi n}(n/e)^n(1+\Theta(1/n))$

$\Rightarrow$ Height of the tree ≥ $n\log n - n\log e = \Omega(n\log n)$

Average Case Complexity

- average depth of leaves = $\dfrac{\sum_{v \text{ leaf}} depth(v)}{\# \text{ leaves}}$

- Among all binary trees with L leaves, $\sum_{v \text{ leaf}} depth(v)$

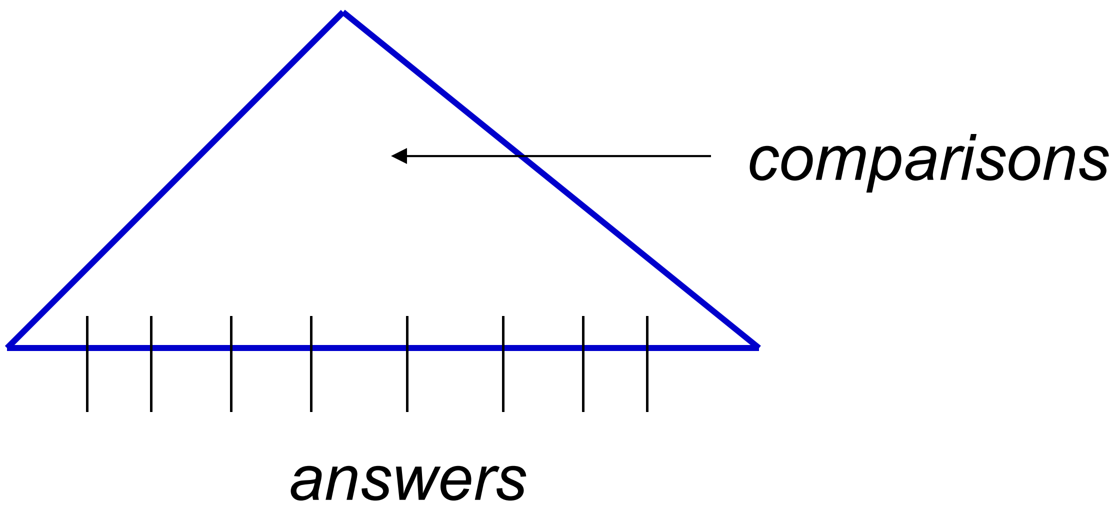
(="external path length") minimized by full binary tree where all leaves at same or adjacent levels

Proof: Interchange argument otherwise reduces external path length

$\Rightarrow$ Average case also Ω(nlogn)

Decision Trees for other problems

- Maximum, minimum, selection, ...
- Search problem $x \in S$?



- For any problem, log(#answers) is a lower bound
- but sometimes not a tight lower bound

Adversarial lower bound

- Example: Maximum needs *n-1* comparisons
- # possible answers = n
- log(# possible answers)= log*n* : bound too weak

- ∀ input, all elements except maximum must lose a comparison, otherwise adversary can change the input and force wrong answer
- Progress from initial state : # losers = 0

  to final state # losers = n-1

  ⇒ # comparisons ≥ n-1

Other problems

- Search Problem $x \in S$?
  - Unsorted S: n comparisons
  - Sorted S: $\log n$

- Duplicate elements?: $\Omega(n \log n)$
- Set Operations (∪, ∩, −): $\Omega(n \log n)$

- Simultaneous max and min: $\lceil 3n/2 \rceil - 2$

(HW exercise)