Chapter 22

Lower bounds

NEW CS 473: Theory II, Fall 2015
November 12, 2015

22.1 Sorting

22.1.0.1 Sorting...

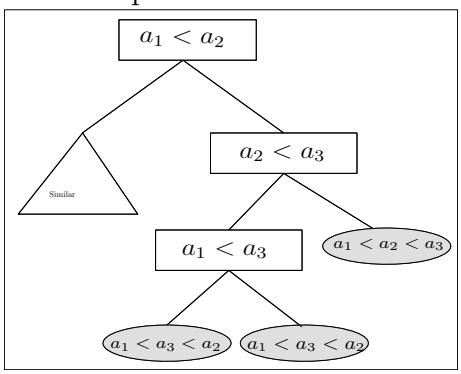
(A) n items: $x_1, \ldots, x_n$.
(B) Can be sorted in $O(n \log n)$ time.
(C) Claim: $\Omega(n \log n)$ time to solve this.
(D) Rules of engagement: What can an algorithm do???

22.1.0.2 Comparison model

(A) In the ***comparison model***:
  (A) Algorithm only allowed to compare two elements.
  (B) **compare**(i, j): Compare ith item in input to jth item in input.
(B) **Q:** # calls to **compare** a deterministic sorting algorithm has to perform?

22.1.0.3 Decision tree for sorting

(A) sorting algorithm: a decision procedure.
(B) Each stage: has current collection of comparisons done.
(C) ... need to decide which comparison to perform next.



22.1.0.4 Sorting algorithm...

(A) sorting algorithm outputs a permutation.
(B) ... order of the input elements so sorted.
(C) Example: Input $x_1 = 7, x_2 = 3, X_3 = 1, x_4 = 19, x_5 = 2$.
  (A) Output: $1, 2, 3, 7, 19$.
  (B) Output: x_3, x_5, x_2, x_1, x_4.
  (C) Output: $\pi = (3, 5, 2, 1, 4)$
  (D) Output as permutation: $\pi(1) = 3, \pi(2) = 5, \pi(3) = 2, \pi(4) = 1, \pi(5) = 4$.
(D) **Interpretation**: $x_{\pi(i)}$ is the ith smallest number in $x_1, \ldots, x_n$.
(E) v: Node of decision tree.
  $P(v)$: A set of all permutations compatible with the set of comparisons from root to v.

22.1.0.5 What are permutations?

(A) $\pi = (3, 4, 1, 2)$ is permutation in $P(v)$.
(B) Formally $\pi : [\![n]\!] \to [\![n]\!]$ is a one-to-one function.
  $[\![n]\!] = \{1, \ldots, n\}$
  can be written as:
  $\pi = (3, 4, 1, 2) = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 4 & 1 & 2 \end{pmatrix}$
(C) Input is: x_1, x_2, x_3, x_4
(D) If arrived to v and $\pi \in P(v)$ then
$$x_3 < x_4 < x_1 < x_2.$$
  a possible ordering (as far as what seen so far).
(E)

22.1.0.6 Input realizing a permutation, by example

(A) Let $\pi = (3, 4, 2, 1) = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 4 & 2 & 1 \end{pmatrix}$
(B) Then the input $\pi^{-1} = (3, 4, 1, 2) = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 1 & 2 \end{pmatrix}$
(C) ... would generate this permutation.
(D) Formally
  $x_1 = \pi^{-1}(1) = 4 \ldots x_i = \pi^{-1}(i) \ldots$

22.1.0.7 Back to sorting...

(A) v: a node in decision tree.
(B) If $|P(v)| > 1$: more than one permutation associated with it...
(C) algorithm must continue performing comparisons
(D) ...otherwise, not know what to output...
(E) **Q:** What is the worst running time of algorithm?
(F) Answer: Longest path from root in the decision tree.
  ...because we count only comparisons!

22.1.0.8 Lower bound on sorting...

Lemma 22.1.1. *Any deterministic sorting algorithm in the comparisons model, must perform $\Omega(n \log n)$ comparisons.*

Proof
(A) Algorithm in the comparison model ≡ a decision tree.
(B) Use an adversary argument.
(C) Adversary pick the worse possible input for the algorithm.
(D) Input is a permutation.
(E) $\mathcal{T}$: the optimal decision tree.
(F) $|P(r)| = n!$, where $r = \text{root}(\mathcal{T})$.

22.1.0.9 Proof continued...

(A) u, v: children of r.
(B) Adversary: no commitment on which of the permutations of $P(r)$ it is using.
(C) Algorithm perform compares x_i to x_j in root...
(D) Adversary computes $P(u)$ and $P(v)$
[Adversary has infinite computation power!]
(E) Adversary goes to u if $|P(u)| \geq |P(v)|$, and to v otherwise.
(F) Adversary traversal: always pick child with more permutations.
(G) $v_1, \ldots, v_k$: path taken by adversary.
(H) Adversary input:
The input realizing the single permutation of $P(v_k)$.

22.1.0.10 Proof continued...

(A) Note, that

$$1 = |P(v_k)| \geq \frac{|P(v_{k-1})|}{2} \geq \ldots \geq \frac{|P(v_1)|}{2^{k-1}}.$$

(B) $2^{k-1} \geq |P(v_1)| = n!$.
(C) $k \geq \lg(n!) + 1 = \Omega(n \log n)$.
(D) Depth of $\mathcal{T}$ is $\Omega(n \log n)$. ■

22.2 Uniqueness

22.2.1 Uniqueness

22.2.1.1 Uniqueness

Problem 22.2.1. Given an input of n real numbers $x_1, \ldots, x_n$. Decide if all the numbers are unique.

(A) Intuitively: easier than sorting.
(B) Can be solved in linear time!
(C) ...but in a strange computation model.
(D) Surprisingly...

Theorem 22.2.2. *Any deterministic algorithm in the comparison model that solves Uniqueness, has $\Omega(n \log n)$ running time in the worst case.*

(E) Different models, different results.

22.2.1.2 Uniqueness lower bound

Proof similar but trickier.

$\mathcal{T}$: decision tree (every node has three children).

Lemma 22.2.3. *v: node in decision tree. If $P(v)$ contains more than one permutation, then there exists two inputs which arrive to v, where one is unique and other is not.*

Proof

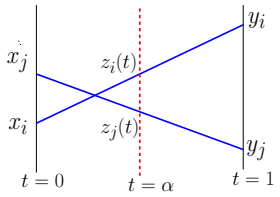
(A) σ, σ': any two different permutations in $P(v)$.
(B) $X = x_1, \ldots, x_n$ be an input realizing σ.
(C) $Y = y_1, \ldots, y_n$: input realizing σ'.
(D) Let $Z(t) = (z_1(t), \ldots, z_n(t))$ an input where $z_i(t) = tx_i + (1-t)y_i$, for $t \in [0,1]$.

22.2.1.3 Proof continued...

(A) $Z(t) = (z_1(t), \ldots, z_n(t))$ an input where $z_i(t) = tx_i + (1-t)y_i$, for $t \in [0,1]$.
(B) $Z(0) = (x_1, \ldots, x_n)$ and $Z(1) = (y_1, \ldots, y_n)$.
(C) Claim: $\forall t \in [0,1]$ the input $Z(t)$ will arrive to the node v in $\mathcal{T}$.

22.2.1.4 Proof of claim...

(A) Assume false.
(B) Assume for $t = \alpha \in [0,1]$ the input $Z(t)$ did not get to v in $\mathcal{T}$.
(C) Assume: compared the ith to jth input element, when paths diverted in $\mathcal{T}$.
(D) I.e., Different path in $\mathcal{T}$ then the one for X and Y.
(E) Claim: $x_i < x_j$ and $y_i > y_j$ or $x_i > x_j$ and $y_i < y_j$.
(F) In either case X or Y will not arrive to v in $\mathcal{T}$.
(G) Consider the functions $z_i(t)$ and $z_j(t)$:

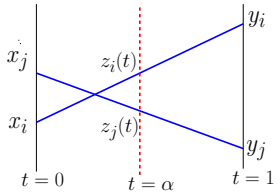


22.2.1.5 Proof of claim continued...

(A) Ordering between $z_i(t)$ and $z_j(t)$ is either ordering between x_i and x_j or the ordering between y_i and y_j.
(B) Conclusion: $\forall t$: inputs $Z(t)$ arrive to the same node $v \in \mathcal{T}$. ■

22.2.1.6 Back to proof of Lemma...

(A) Recap:
  (A) Recall: X, Y to different permutations that their distinct input arrives to the same node $v \in \mathcal{T}$.
  (B) Proved: $\forall t \in [0,1]$: $Z(t) = (z_1(t), \ldots, z_n(t))$ arrives to same node $v \in \mathcal{T}$.
(B) However: There must be $\beta \in (0,1)$ where $Z(\beta)$ has two numbers equal:



(C) $Z(\beta)$: has a pair of numbers that are not unique.

22.2.1.7 Proof of Lemma continued...

(A) Done: Found inputs $Z(0)$ and $Z(\beta)$
(B) such that one is unique and the other is not.
(C) ... both arrive to v. ■

Proved the following:

Lemma 22.2.4. *v: node in decision tree. If $P(v)$ contains more than one permutation, then there exists two inputs which arrive to v, where one is unique and other is not.*

22.2.1.8 Uniqueness takes $\Omega(n \log n)$ time

(A) Apply the same argument as before.
(B) If in the decision tree, the adversary arrived to a node...
(C) containing more than one permutation, it continues into the child with more permutations.
(D) As in the sorting argument, it follows that there exists a path in $\mathcal{T}$ of length $\Omega(n \log n)$.
(E) We conclude:

Theorem 22.2.5. *Solving* **Uniqueness** *for a set of n real numbers takes $\Theta(n \log n)$ time in the comparison model.*

22.2.2 Algebraic tree model

22.2.2.1 Algebraic tree model

(A) At each node, allowed to compute a polynomial, and ask for its sign at a certain point
(B) Example: comparing x_i to x_j is equivalent to asking if the polynomial $x_i - x_j$ is positive/negative/zero).
(C) One can prove things in this model, but it requires considerably stronger techniques.

Problem 22.2.6. (Degenerate points) Given a set P of n points in $\mathbb{R}^d$, deciding if there are $d+1$ points in P which are co-linear (all lying on a common plane).

(D) Jeff Erickson and Raimund Seidel: Solving the degenerate points problem requires $\Omega(n^d)$ time in a "reasonable" model of computation.

22.3 3Sum-Hard

22.3.1 3Sum-Hard

22.3.1.1 3Sum-Hard

(A) Consider the following problem:

Problem 22.3.1. (3SUM): Given three sets of numbers A, B, C are there three numbers $a \in A$, $b \in B$ and $c \in C$, such that $a + b = c$.

(B) One can show...

Lemma 22.3.2. *One can solve the 3SUM problem in $O(n^2)$ time.*

Proof: Exercise... ■

22.3.1.2 3Sum-Hard continued

(A) Somewhat surprisingly, no better solution is known.
(B) Open Problem: Find a subquadratic algorithm for **3SUM**.
(C) It is widely believed that no such algorithm exists.
(D) There is a large collection problems that are 3SUM-Hard: if you solve them in subquadratic time, then you can solve 3SUM in subquadratic time.

22.3.1.3 3SUM-hard problems

(A) Those problems include:
    (a) For n points in the plane, is there three points that lie on the same line.
    (b) Given a set of n triangles in the plane, do they cover the unit square
    (c) Given two polygons P and Q can one translate P such that it is contained inside Q?
(B) So, how does one prove that a problem is 3SUM hard?
(C) Reductions.
(D) Reductions must have subquadratic running time.
(E) The details are interesting, but are omitted.