Lower Bound for Algorithms

* review

Big-Omega:

\[ f(n) \text{ is } \Omega(g(n)) \text{ if } \]

\[ c \cdot g(n) \leq f(n) \text{ for constant } c \text{ and } n > n_0 \]

\[ \Rightarrow \]

* Let \( T(A, f) \) be the number of functional evaluations performed by some algo \( A \) given the step function \( f \) as input

* "Time complexity of Algo \( A \)" is the worst-case running time over all the possible inputs

\[ T(A) = \max_{|f|=n} T(A, f) \]

* "Time complexity of the problem" is time complexity of the fastest algo that correctly solves it.
\[ T(n) = \min_A T(A) = \min_{f \leq n} \max_A T(A, f) \]

1. Comparison based Algo

Algos can be represented as Decision Trees:
- rooted, ordered, binary.

- internal node: step function
- leaf: output algo.

Execution of algo gives path from root to a leaf (deterministic).

C(x) d.t. for sorting \( n = 3 \)
- \( \text{Input} = [a, b, c] \)

- \( \text{Theorem} \)
  - Any decision tree that sorts \( n \) distinct elements has height at least \( \log_2(n!) \).
proof

- Since the answer can be any of $N!$ permutations of the input, with at least $N!$ leaves.
- Since there are at most 2 children of any non-leaf node, a tree of height $h$ can have at most $2^h$ leaves.

Thus, the height must be at least $\log_2 N!$.

Corollary

Any algorithm that sorts $n$ elements by comparison requires $\Omega(n \log n)$ time for some input of size $n$.

Proof

Any comparison-based algorithm can be modeled by a decision tree. There is going to be some leaf at least $\log_2 (n!)$.

Thus, there is a computation that will perform at least $\log_2 (n!)$ of comparisons.
\[\log (n!) \geq \log (n^{n/2}) = \frac{n}{2} \log \frac{n}{2} = O(n \log n)\]

\[\Omega (n \log n)\]

- **Algebraic Computational Tree (ACT)**
  - Input values \( v = \{x_1, \ldots, x_n\} \)
  - Non-leaf node \( v \):
    - Are either with 1) one child
      - Which contains instructions
        - \( f_v \rightarrow f_v, \) for \( o \in \{+, -, \times, \sqrt{\cdot}\} \)
        - \( f_v \rightarrow c \)
        - \( f_v \rightarrow f_v; \)
        - \( f_v \rightarrow x_i \)
  - 2) with 2 children which returns
    - \( f_v > 0 \quad f_v < 0 \quad f_v = 0 \) (yes or no)
  - Leaf nodes: labeled with binary strings
If it is Y/N answer, then 0 or 1

If it is combinatorial answer:

bring string that encodes the information output

\[ \text{Claim:}\] sorting takes \( \Omega(n \log n) \) time in AIT model.

Proof:
there must be at least \( n! \) leaves
for all of the input permutations.
The \# of children of the node is \( \text{at most 2} \). Thus the height of the tree must be at least \( \lg(n!) \), as before in decision trees.

* Problem Reduction

Given 2 problems A and B, A is \( f(n) \)-transformable to B if
1) any instance of A can be transformed into an instance of B in \( O(f(n)) \) time
2) the solution for B can be transformed into a solution of A in \( O(f(n)) \) time.
where \( n \) is the size of the problem.

Let \( A \) be \( f(n) \)-transformable to \( B \).

1) If \( A \) requires \( g(n) \) time, then \( B \) takes at least \( g(n) - c'f(n) \) time.

\[
\begin{aligned}
A & \quad \xrightarrow{f(n) - tm} \quad B \\
\xlongleftarrow{g(n)} & \quad \xrightarrow{g(n) - c'f(n)}
\end{aligned}
\]

2) If \( B \) is solvable in \( h(n) \) time, then \( A \) is solvable in \( c'f(n) + h(n') \) time, for some constant \( c' \) and \( n' \) the size of transformed input.

\[
\begin{aligned}
A & \quad \xrightarrow{f(n) - tm} \quad B \\
\xlongleftarrow{c'f(n) + h(n)} & \quad \xrightarrow{h(n')} 
\end{aligned}
\]

Problem: Rectilinear Steiner tree

Given a set of points in the plane, find a set of horizontal and vertical line segments of minimal total length that connects all the points.
• $O(n)$ time reduction from sorting to $A ightarrow RST \rightarrow B$

$O(n)$

1. sorting input: $Ev_1, \ldots, V_n$
2. transform input

\[ p_i = (Ev_i, 0) \text{ for } i = 1 \ldots n \]

$O(n)$

\[ \Rightarrow \text{Given the solution to } RST \rightarrow \]

$O(n)$

\[ \text{Search the solution starting from the left most end point and print of } \]

\[ V_i \text{ when each point } p_i \text{ is encountered} \]

\[ g(n) - c(f(n)) \]

\[ \sum (n1n) \rightarrow RST \sum (n1n) \]

Problem 2: Maximum Empty Convex Region Problem

Given a rectangle in the plane and $n$ points inside the rectangle,
Find the largest convex region on the rectangle that contains no points and output the boundary of the region in counter clockwise order.

Lower Bound using sorting:
Reduce from sorting n positive value $v_1, \ldots, v_n$ to MEKR

1. Input transform.

Let $m = \max \{v_1, \ldots, v_n\}$

@ $p_i = \left(\frac{m + v_i}{2}, \frac{v_i^2}{2}\right)$

$\Theta(n)$

\[
\begin{cases}
    p_{11} = (0, 0) \\
    p_{12} = (0, 2m^2) \\
    p_{13} = (2m, 2m^2)
\end{cases}
\]

2. Output transformation

$\Theta(n)$ to translate solution of MEKR start at $(0,0)$ and walk around the region output the corresponding $v_i$ values.

Since reduction takes $\Theta(n)$ time but we have $\Omega(n \log n)$ for sorting.
the lower bound of \( \text{MECR} \)

\[
g(n) - c\log n = \Omega(n \log n) - c\Theta(n)
\]

\[
\therefore \Omega(n \log n)
\]

*other problems that has \( \Omega(n \log n) \)

in ACT model

1) element uniqueness problem

- given a set of numbers, determine if the elements are distinct

2) set equality on inclusion

- given set \( A = \{x_1, \ldots, x_n\} \)

- set \( B = \{y_1, \ldots, y_n\} \)

- determine if \( A = B \) or \( A \subseteq B \)

3) set disjointness P.

- determine if \( A \cap B = \emptyset \)