What You Should Know About Algorithm Design

Hengfeng Wei

hfwei@nju.edu.cn

April 19, 2018


As
Fast
As
Possible


Jamaica
2163
USA
3215


Jamaica
2163
USA
3215


KNOW
YOUR
LIMITS


Jamaica
2163
3215


KNOW
YOUR
LIMITS


快过闪电
博尔特自传
USAIN BOLT

Problem P Algorithm A

Inputs: $\mathcal{X}_n$ of size n

Problem P Algorithm A

Inputs: $\mathcal{X}_n$ of size n

$$W_A(n) = \max_{x \in \mathcal{X}_n} T_A(x)$$

$$B_A(n) = \min_{x \in \mathcal{X}_n} T_A(x)$$

$$A_A(n) = \sum_{x \in \mathcal{X}_n} T_A(x) \cdot P(x) = \mathbb{E}[T_A] = \sum_{t \in T_A(\mathcal{X}_n)} t \cdot P(T = t)$$

Problem P Algorithm A

Inputs: $\mathcal{X}_n$ of size n

$$W_A(n) = \max_{x \in \mathcal{X}_n} T_A(x)$$

$$B_A(n) = \min_{x \in \mathcal{X}_n} T_A(x)$$

$$A_A(n) = \sum_{x \in \mathcal{X}_n} T_A(x) \cdot P(x) = \mathbb{E}[T_A] = \sum_{t \in T_A(\mathcal{X}_n)} t \cdot P(T = t)$$

$$T_P(n) =$$

Problem P Algorithm A

Inputs: $\mathcal{X}_n$ of size n

$$W_A(n) = \max_{x \in \mathcal{X}_n} T_A(x)$$

$$B_A(n) = \min_{x \in \mathcal{X}_n} T_A(x)$$

$$A_A(n) = \sum_{x \in \mathcal{X}_n} T_A(x) \cdot P(x) = \mathbb{E}[T_A] = \sum_{t \in T_A(\mathcal{X}_n)} t \cdot P(T = t)$$

$$T_P(n) = \min_{A \text{ solves } P} W_A(n)$$

Problem P Algorithm A

Inputs: $\mathcal{X}_n$ of size n

$$W_A(n) = \max_{x \in \mathcal{X}_n} T_A(x)$$

$$B_A(n) = \min_{x \in \mathcal{X}_n} T_A(x)$$

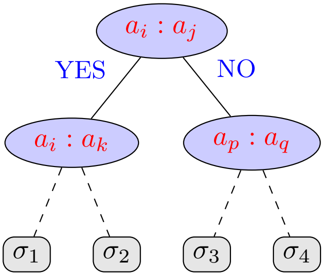
$$A_A(n) = \sum_{x \in \mathcal{X}_n} T_A(x) \cdot P(x) = \mathbb{E}[T_A] = \sum_{t \in T_A(\mathcal{X}_n)} t \cdot P(T = t)$$

$$T_P(n) = \min_{A \text{ solves } P} W_A(n) = \min_{A \text{ solves } P} \max_{x \in \mathcal{X}_n} T_A(x)$$

Algorithm Designer

Algorithm Designer

Lower Bound Prover



Decision Trees

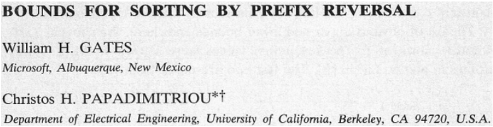
Adversary Argument

Lower Bound for Comparison-based Sorting

Prove a lower bound of $\Omega(n \log n)$ on the time complexity of any **comparison-based** sorting algorithm on inputs of size n.

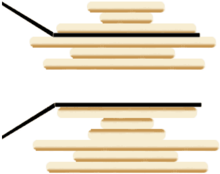
Lower Bound for Comparison-based Sorting

Prove a lower bound of $\Omega(n \log n)$ on the time complexity of any **comparison-based** sorting algorithm on inputs of size n.


BOUNDS FOR SORTING BY PREFIX REVERSAL

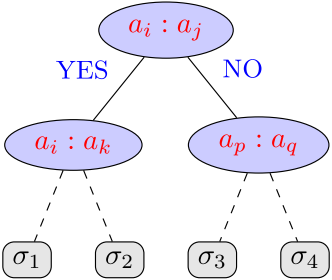
William H. GATES
Microsoft, Albuquerque, New Mexico

Christos H. PAPADIMITRIOU*†
Department of Electrical Engineering, University of California, Berkeley, CA 94720, U.S.A.



Decision Tree Model

Decision Tree Model



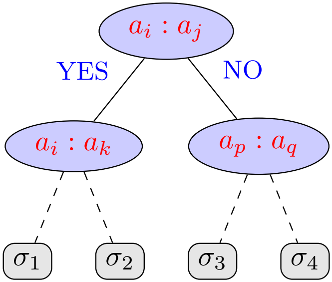
Nodes: comparisions $a_i : a_j$

$$<,\ \leq,\ =,\ \geq,\ >$$

Edges: two-way decisions (Y/N)

Leaves: possible permutations

Decision Tree Model



Nodes: comparisions $a_i : a_j$

$$<, \leq, =, \geq, >$$

Edges: two-way decisions (Y/N)

Leaves: possible permutations

Assumption:

All the input elements are **distinct**.

$$a_i < a_j$$

Decision Tree Model

$$\boxed{\text{Any Comparison-based Sorting Algorithm} \xRightarrow{\text{modeled by}} \text{A Decision Tree}}$$

Decision Tree Model

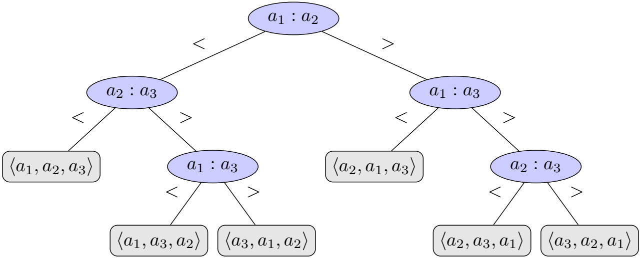
Any Comparison-based Sorting Algorithm $\xRightarrow{\text{modeled by}}$ A Decision Tree

Decision Tree Model

$$\boxed{\text{Any Comparison-based Sorting Algorithm} \xRightarrow{\textsf{modeled by}} \text{A Decision Tree}}$$

Decision Tree Model

Any Comparison-based Sorting Algorithm $\xrightarrow{\text{modeled by}}$ A Decision Tree



The decision tree for insertion sort on three elements.

Decision Tree Model

Any Comparison-based Sorting Algorithm $\xrightarrow{\text{modeled by}}$ A Decision Tree

---

procedure -SORT(A, n)
    for $i \gets 1$ **to** $n - 1$ **do**
        for $j \gets i + 1$ **to** n **do**
            if $A[j] < A[i]$ **then**
                SWAP$(A[j], A[i])$

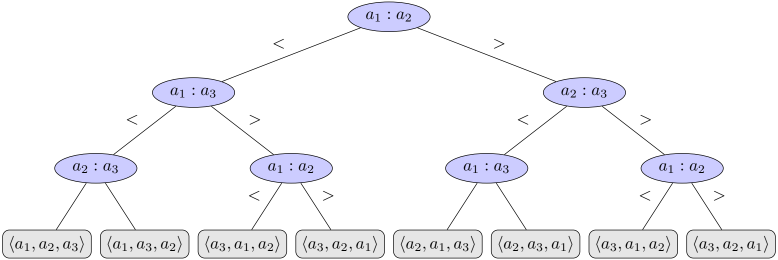
Decision Tree Model

Any Comparison-based Sorting Algorithm $\xRightarrow{\text{modeled by}}$ A Decision Tree

```
procedure Selection-Sort(A, n)
    for i ← 1 to n − 1 do
        for j ← i + 1 to n do
            if A[j] < A[i] then
                Swap(A[j], A[i])
```

Decision Tree Model

Any Comparison-based Sorting Algorithm $\xRightarrow{\text{modeled by}}$ A Decision Tree



The decision tree for selection sort on three elements.

Decision Tree Model

Any Comparison-based Sorting Algorithm $\mathcal{A} \xrightarrow{\text{modeled by}}$ A Decision Tree $\mathcal{T}$

Decision Tree Model

$\boxed{\text{Any Comparison-based Sorting Algorithm } \mathcal{A} \xRightarrow{\text{modeled by}} \text{A Decision Tree } \mathcal{T}}$

Algorithm $\mathcal{A}$ on a specific input of size n $\xRightarrow{\text{modeled by}}$ A path through $\mathcal{T}$

Decision Tree Model

$$\boxed{\text{Any Comparison-based Sorting Algorithm } \mathcal{A} \xRightarrow{\text{modeled by}} \text{A Decision Tree } \mathcal{T}}$$

$$\text{Algorithm } \mathcal{A} \text{ on a specific input of size } n \xRightarrow{\text{modeled by}} \text{A path through } \mathcal{T}$$

$$\text{Worst-case time complexity of } \mathcal{A} \xRightarrow{\text{modeled by}} \text{The height of } \mathcal{T}$$

Decision Tree Model

$$\boxed{\text{Any Comparison-based Sorting Algorithm } \mathcal{A} \xRightarrow{\text{modeled by}} \text{A Decision Tree } \mathcal{T}}$$

$$\text{Algorithm } \mathcal{A} \text{ on a specific input of size } n \xRightarrow{\text{modeled by}} \text{A path through } \mathcal{T}$$

$$\text{Worst-case time complexity of } \mathcal{A} \xRightarrow{\text{modeled by}} \text{The height of } \mathcal{T}$$

Worst-case Lower Bound of Comparison-based Sorting
(on inputs of size n)

$$\xRightarrow{\text{modeled by}}$$

The Minimum Height of All $\mathcal{T}$s

Worst-case Lower Bound of Comparison-based Sorting
(on inputs of size n)

$$\xRightarrow{\text{modeled by}}$$

The Minimum Height of All $\mathcal{T}$s

Worst-case Lower Bound of Comparison-based Sorting
(on inputs of size n)

$$\xRightarrow{\text{modeled by}}$$

The Minimum Height of All $\mathcal{T}$s

To be a correct sorting algorithm:

$$\#\text{ of leaves } \geq n!$$

Worst-case Lower Bound of Comparison-based Sorting
(on inputs of size n)

$$\xRightarrow{\text{modeled by}}$$

The Minimum Height of All $\mathcal{T}$s

To be a correct sorting algorithm:

$$\text{\# of leaves } \geq n!$$

To be a full binary tree:

$$\text{\# of leaves } \leq 2^h$$

Lower Bound for Comparison-based Sorting

$$\boxed{n! \le \# \text{ of leaves} \le 2^h}$$

Lower Bound for Comparison-based Sorting

$$\boxed{n! \leq \# \text{ of leaves} \leq 2^h}$$

$$h \geq \log n! = \Omega(n \log n)$$

Lower Bound for Comparison-based Sorting

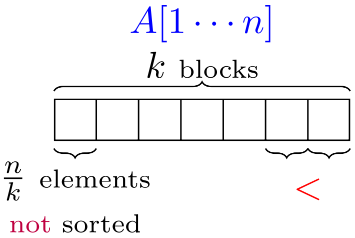
$$\boxed{n! \le \# \text{ of leaves} \le 2^h}$$

$$h \ge \log n! = \Omega(n \log n)$$

Stirling Formula (by *James Stirling*):

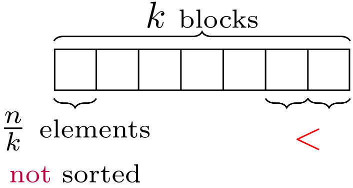
$$n! = \Theta\left(\sqrt{2\pi n}\left(\frac{n}{e}\right)^n\right)$$

Definition (K-sorting (Problem 6.8))



Definition (K-sorting (Problem 6.8))

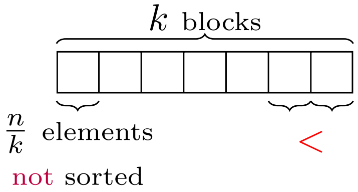
$$A[1 \cdots n]$$



$$O(n \log k)$$

Definition (K-sorting (Problem 6.8))

$$A[1 \cdots n]$$



$$O(n \log k)$$

$$n = 16,\ k = 4,\ \frac{n}{k} = 4$$

1, 2, 4, 3;   7, 6, 8, 5;   10, 11, 9, 12;   15, 13, 16, 14

k-sorted

k-sorted

1-sorted

k-sorted

1-sorted $\rightarrow$ 2-sorted

k-sorted

1-sorted $\rightarrow$ 2-sorted $\rightarrow$ 4-sorted

k-sorted

$$1\text{-sorted} \to 2\text{-sorted} \to 4\text{-sorted} \to \cdots \to n\text{-sorted}$$

k-sorted

1-sorted $\to$ 2-sorted $\to$ 4-sorted $\to \cdots \to$ n-sorted

Quicksort (with median as pivot) stops after the $\log k$ recursions.

k-sorted

$$1\text{-sorted} \to 2\text{-sorted} \to 4\text{-sorted} \to \cdots \to n\text{-sorted}$$

Quicksort (with median as pivot) stops after the $\log k$ recursions.

$$\boxed{\Theta(n \log k)}$$

$$\Omega(n \log k)$$

$$\Omega(n \log k)$$

$L \geq$

$$\Omega(n \log k)$$

$$L \geq \binom{n}{n/k}\binom{n - n/k}{n/k}\cdots\binom{n/k}{n/k}$$

$$\Omega(n \log k)$$

$$L \geq \binom{n}{n/k}\binom{n-n/k}{n/k}\cdots\binom{n/k}{n/k} = \binom{n}{n/k,\ldots,n/k}$$

$$\Omega(n \log k)$$

$$L \geq \binom{n}{n/k}\binom{n - n/k}{n/k}\cdots\binom{n/k}{n/k} = \binom{n}{n/k, \ldots, n/k} = \frac{n!}{\left((\frac{n}{k})!\right)^k}$$

$$\Omega(n \log k)$$

$$L \geq \binom{n}{n/k}\binom{n - n/k}{n/k} \cdots \binom{n/k}{n/k} = \binom{n}{n/k, \ldots, n/k} = \frac{n!}{\left((\frac{n}{k})!\right)^k}$$

$$H \geq \log\left(\frac{n!}{\left((\frac{n}{k})!\right)^k}\right)$$

$$\Omega(n \log k)$$

$$L \geq \binom{n}{n/k}\binom{n-n/k}{n/k}\cdots\binom{n/k}{n/k} = \binom{n}{n/k,\ldots,n/k} = \frac{n!}{\left((\frac{n}{k})!\right)^k}$$

$$H \geq \log\left(\frac{n!}{\left((\frac{n}{k})!\right)^k}\right) = \Omega(n \log k)$$

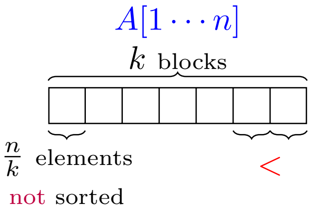
$$\Omega(n \log k)$$

$$L \geq \binom{n}{n/k}\binom{n-n/k}{n/k}\cdots\binom{n/k}{n/k} = \binom{n}{n/k,\ldots,n/k} = \frac{n!}{\left((\frac{n}{k})!\right)^k}$$

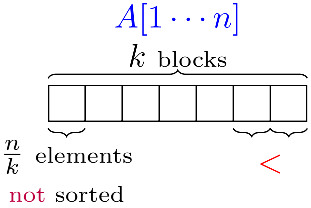
$$H \geq \log\left(\frac{n!}{\left((\frac{n}{k})!\right)^k}\right) = \Omega(n \log k)$$

$$\boxed{n! \sim \sqrt{2\pi n}\left(\frac{n}{e}\right)^n \implies \log n! \sim n \log n}$$

Sorting the k-sorted array



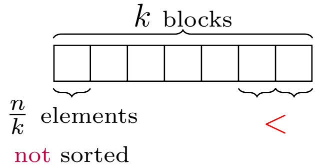
Sorting the k-sorted array



$$O(n \log \frac{n}{k})$$

Sorting the k-sorted array

$A[1 \cdots n]$

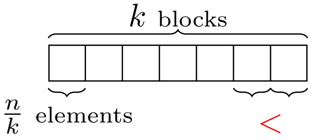


$L \geq ((\frac{n}{k})!)^k$

$O(n \log \frac{n}{k})$

Sorting the k-sorted array

$$A[1 \cdots n]$$



not sorted

$$L \geq ((\frac{n}{k})!)^k$$

$$O(n \log \frac{n}{k})$$

$$H \geq \log((\frac{n}{k})!)^k = \Omega(n \log \frac{n}{k})$$

Bolts and Nuts (Problem 6.9)

Bolts and Nuts (Problem 6.9)

Quicksort

Bolts and Nuts (Problem 6.9)

Quicksort

$$A(n) = O(n \log n)$$

Bolts and Nuts (Problem 6.9)

Quicksort

$$A(n) = O(n \log n)$$

In the worst case:

- "Matching Nuts and Bolts" by Alon *et al.*, $\Theta(n \log^4 n)$
- "Matching Nuts and Bolts Optimality" by Bradford, 1995, $\Theta(n \log n)$

$$\Omega(n \log n)$$

$$\Omega(n \log n)$$

$$3^H \geq L \geq n!$$

$$\Omega(n \log n)$$

$$3^H \geq L \geq n! \implies H \geq \log n! \implies H = \Omega(n \log n)$$

Repeated Elements (Problem 6.13)

$$R[1 \ldots n]$$

$$\# > \lfloor \frac{n}{13} \rfloor$$

To find all $\dfrac{n}{13}$-repeated elements

$\textsc{Check}(R[i], R[j])$

$$\# > \lfloor \frac{n}{k} \rfloor$$

$$\# > \lfloor \frac{n}{k} \rfloor$$

$$\Omega(n \log k)$$

$$\# > \lfloor \frac{n}{k} \rfloor$$

$$\Omega(n \log k)$$

Searching in Matrix (Problem 9.8)

$$M : m \times n$$

Row: increasing from left to right

Col: increasing from top to down

$$x \in M?$$

Searching in Matrix (Problem 9.8)

$$M : m \times n$$

Row: increasing from left to right

Col: increasing from top to down

$$x \in M?$$

Divide & Conquer

Searching in Matrix (Problem 9.8)

$$M : m \times n$$

Row: increasing from left to right

Col: increasing from top to down

$$x \in M?$$

Divide & Conquer

$$T(m,n) = 3T(\frac{m}{2}, \frac{n}{2}) + 1$$

Searching in Matrix (Problem 9.8)

$$M : m \times n$$

Row: increasing from left to right

Col: increasing from top to down

$$x \in M?$$

Divide & Conquer

$$T(m, n) = 3T(\frac{m}{2}, \frac{n}{2}) + 1$$

Always checking the lower left corner.

Searching in Matrix (Problem 9.8)

$$M : m \times n$$

Row: increasing from left to right

Col: increasing from top to down

$$x \in M?$$

Divide & Conquer

$$T(m,n) = 3T(\frac{m}{2}, \frac{n}{2}) + 1$$

Always checking the lower left corner.

$$T(m,n) = m + n - 1$$

Assume $M : n \times n$

$$W(n) \leq 2n - 1$$

Assume $M : n \times n$

$$W(n) \leq 2n - 1$$

$$\boxed{W(n) \geq 2n - 1}$$

Assume $M : n \times n$

$$W(n) \leq 2n - 1$$

$$\boxed{W(n) \geq 2n - 1}$$

By Adversary Argument!

Assume $M : n \times n$

$$W(n) \leq 2n - 1$$

$$\boxed{W(n) \geq 2n - 1}$$

By Adversary Argument!

Diagonals: $i + j = n - 1 \quad \& \quad i + j = n$

$$\text{Assume } M : n \times n$$

$$W(n) \leq 2n - 1$$

$$\boxed{W(n) \geq 2n - 1}$$

By Adversary Argument!

$$\text{Diagonals: } i + j = n - 1 \quad \& \quad i + j = n$$

No particular ordering requirements on these two diagonals!

Assume $M : n \times n$

$$W(n) \leq 2n - 1$$

$$\boxed{W(n) \geq 2n - 1}$$

By Adversary Argument!

Diagonals: $i + j = n - 1 \quad \& \quad i + j = n$

No particular ordering requirements on these two diagonals!

$$i + j \leq n - 1 \implies x > M_{ij}$$
$$i + j > n - 1 \implies x < M_{ij}$$


Thank
You!

Your opinon Matters

Office 302

Mailbox: H016

hfwei@nju.edu.cn