CS4800: Algorithms & Data
Jonathan Ullman

Lecture 5:

- Divide-and-Conquer: Median-Finding, Binary Search, Max-Difference,…

Jan 23, 2018

Ask the Audience

- You come up with an algorithm for multiplying two n-digit numbers that uses:
  - 6 multiplications of $\frac{n}{4}$-digit numbers
  - $O(n^{3/2})$ time to combine

- Is this algorithm faster than the $\Theta(n^{1.585\ldots})$ time Karatsuba's algorithm?

$T(n) = 6 \times T\left(\frac{n}{4}\right) + n^{3/2}$ $\quad T(n) = a \times T\left(\frac{n}{b}\right) + n^c$ $\quad a = 6$

$b = 4$

$c = 3/2$

$\left(\frac{a}{b^c}\right) = \left(\frac{6}{4^{3/2}}\right) = \frac{6}{8} < 1$

$T(n) = \Theta\left(n^{3/2}\right)$

Selection / Median

Selection

Simple Algorithms: $\Theta(nk)$ (e.g. $\Theta(n^2)$ to find the median)

- Given an array of numbers $A[1, \ldots, n]$, how quickly can I find the k-th smallest number?

11	3	42	28	17	8	2	15

SimpleSelect($A[1, \ldots, n], k$):

  Let T hold the indices of the k smallest items so far

  $T \leftarrow \{1, \ldots, k\}$

$n-k$ For $i = k + 1, \ldots, n$:

    If $A[i]$ is ~~bigger~~ smaller than $A[j]$ for some $j \in T$

    Add i to T and remove j from T

checking takes $\Theta(k)$

  Output the largest item in T

Selection

find the median in $O(n \log n)$ time.

- Selecting the k-th smallest number is no harder than sorting. Takes $\boxed{\Theta(n \log n)}$ time.

11	3	42	28	17	8	2	15

sort

2	3	8	11	15	17	28	42

- Can improve to $\Theta(n \log k)$

Finding the Median

- The **median** is the $\left\lceil \frac{n}{2} \right\rceil$-th smallest number

11	3	42	28	17	8	2	15

$\Downarrow$ sort

2	3	8	11	15	17	28	42

- Today: finding the **median** in $O(n)$ time.

Job Interview Puzzle

- You have 25 horses and want to find the 3 fastest.
- You do not know how fast they are, but you have a racetrack where you can race 5 horses at a time.
  - In: $\{1, 5, 6, 18, 22\}$ Out: $(6 \succ 5 \succ 18 \succ 22 \succ 1)$
- Problem: find the 3 fastest using just 7 races.

Median Algorithm: Take I

17	3	42	11	28	8	2	15	13	A

smaller than the pivot — pivot — larger than the pivot

11	3	15	13	2	8	17	28	42

can do this "partitioning" in $O(n)$ time. ← $A[r]$

QuickSelect$(A[1, \ldots, n], k)$:

If $n = 1$: return $A[1]$ // Base case

$O(n)$ { **Choose a pivot element** $p = A[1]$ E.g. $k=4$, $Select(A[1,\ldots,r-1], k)$; $k=8$, $Select(A[r+1,\ldots,n], 1)$

Partition around the pivot p, let $A[r]$ be the pivot

If $k < r$: return $Select(A[1, \ldots, r-1], k)$

Elif $k > r$: return $Select(A[r+1, \ldots, n], k-r)$

Else: return $A[r]$

recursive call takes $T(r-1)$ or $T(n-r-1)$

Median Algorithm: Take I

17	3	42	11	28	8	2	15	13	A

4th smallest ← 7th

11	3	15	13	2	8	17	~~28~~	~~42~~

A[r]

≥30%

≥30%

Median Algorithm: Take I

$k=9$

pivot

1	2	3	4	5	6	7	8	9	A

pivot

2	3	4	5	6	7	8	9

pivot

3	4	5	6	7	8	9

running time is $\Theta(nk)$ $\quad T(n) \le T(n-1) + O(n)$

For median QuickSelect takes $\Omega(n^2)$ time in the worst case.

Median Algorithm: Take II

- Need to find a pivot element p that is in the "middle" of the sorted list in $O(n)$ time.

- Idea 1: Use the median of this list! "Ideal Pivot"

- Idea 2: Use the "median-of-medians" (MOM)!

Need to find a pivot approximately in the middle

- middle half is good enough

Need to do in $O(n)$ time

Median of Medians (Choosing a Pivot)

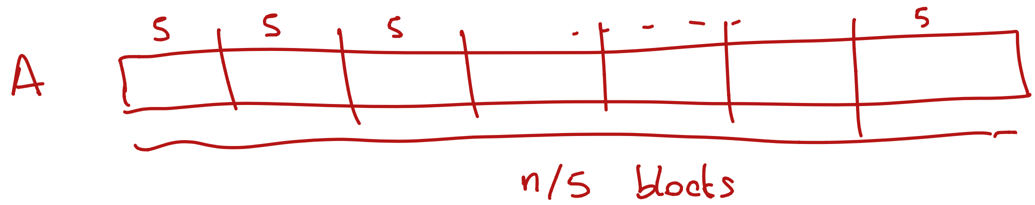
$T_{MOM}(n) = T_{SEL}\left(\frac{n}{5}\right) + O(n)$

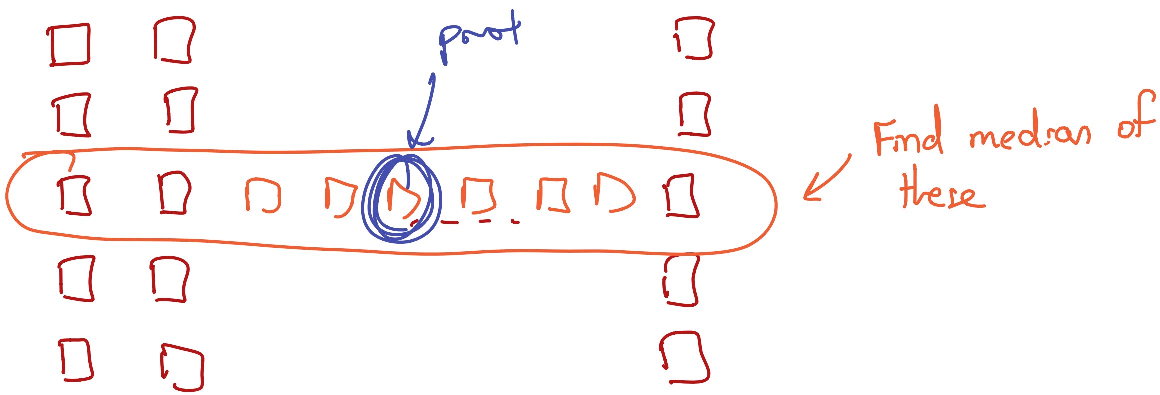
MOM5($A[1, \ldots, n]$):

  $m \leftarrow \lceil n/5 \rceil$

  For $i = 1, \ldots, m$: $M[i] = \text{median}\{A[5i-4, \ldots, 5i]\}$ ← $\frac{n}{5} \times O(1) = O(n)$

  Return $p = \text{QuickSelect}(M[1, \ldots, m], \lfloor m/2 \rfloor)$ ← $T_{SEL}\left(\frac{n}{5}\right)$



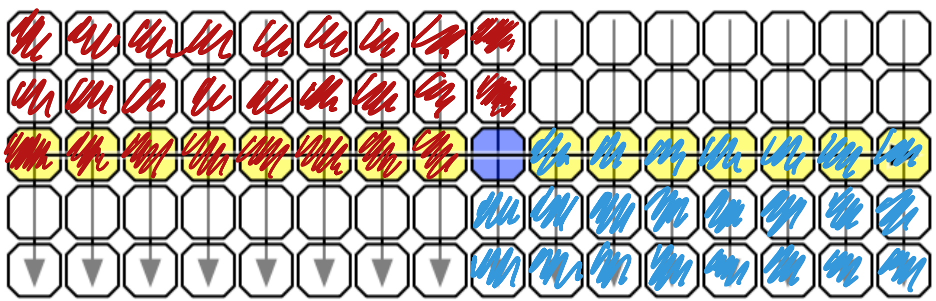


Median of Medians

- Claim: There are at least $3n/10$ items smaller than the MOM and at least $3n/10$ items larger.

= smaller than med

= larger than med



Visualizing the median of medians

$$2 + \left(\frac{n}{10} - 1\right) + \left(2 \times \left(\frac{n}{10} - 1\right)\right)$$

$$\geq \frac{3n}{10} - O(1)$$

$\frac{3n}{10} - O(1)$ are larger

Median Algorithm: Take II

17	3	42	11	28	8	2	15	13

A

11	3	17	13	2	8	17	28	42

QuickSelect($A[1, \ldots, n], k$):

If $n \leq 25$: return ~~med{$A[1, \ldots, n]$}~~ k^{th} smallest

Let $p = \mathrm{MOM5}(A)$ $T_{MOM}(n) = T_{SEL}\left(\frac{n}{5}\right) + O(n)$

Partition around the pivot p, let $A[r]$ be the pivot $O(n)$

If $k < r$: return QuickSelect($A[1, \ldots, r-1], k$) $T(r)$

Elif $k > r$: return QuickSelect($A[r+1, \ldots, n], k - r$) $T(n-r)$

one of these

Else: return $A[r]$

$\leq \max\{T(r), T(n-r)\}$

Median of Medians

Because the MOM is $\geq \frac{3n}{10}$ and $\leq \frac{3n}{10}$

$$T_{SEL}(n) \leq T_{SEL}\left(\frac{7n}{10}\right) + T_{MOM}(n) + C'n$$

$$T_{SEL}(25) \leq C$$

$$\leq T_{SEL}\left(\frac{7n}{10}\right) + T_{SEL}\left(\frac{2n}{10}\right) + \boxed{Cn}$$

for both partitioning and finding the small medians

Median of Medians

elimnate 30%

look at 20%

$$T_{SEL}(n) \le T_{SEL}\left(\frac{7n}{10}\right) + T_{SEL}\left(\frac{2n}{10}\right) + Cn \qquad T_{SEL}(25) \le C$$

Theorem: $\forall n \quad T(n) = O(n)$

work

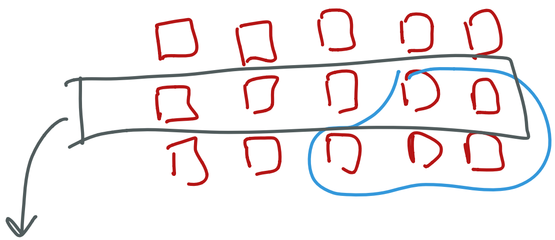
Cn

$C \times \left(\frac{n}{5}\right) + C \times \left(\frac{7n}{10}\right) = C \times \frac{9n}{10}$

$$T(n) = n \times C \times \sum_{i=0}^{l} \left(\frac{9}{10}\right)^i = 10 \times n \times C$$

Ask the Audience

- Suppose we instead split this input into $\frac{n}{3}$ blocks of size 3. Would the algorithm still run in time $O(n)$?



$\#$ smaller is $\frac{n}{6} \times 2 \approx \frac{n}{3}$

$\#$ larger is $\approx \frac{n}{3}$

find the MOM3 $T_{SEL}\left(\frac{n}{3}\right) + O(n)$

$$T_{SEL}(n) \leq T_{SEL}\left(\frac{n}{3}\right) + O(n) + T_{SEL}\left(\frac{2n}{3}\right) + O(n)$$

Median in Practice

17	3	42	11	28	8	2	15	13	A

11	3	17	13	2	8	17	28	42

QuickSelect($A[1, \dots, n], k$):
    If $n = 1$: return $A[1]$

    Let $p = A[k]$ for a **randomly chosen** k
    Partition around the pivot p, let $A[r]$ be the pivot
    If $k < r$: return QuickSelect($A[1, \dots, r-1], k$)
    Elif $k > r$: return QuickSelect($A[r+1, \dots, n], k - r$)
    Else: return $A[r]$

Median in Practice

17	3	42	11	28	8	2	15	13	A

11	3	17	13	2	8	17	28	42

Quicksort

17	3	42	11	28	8	2	15	13	A

11	3	17	13	2	8	17	28	42

recursively sort

recursively

QuickSort($A[1, \dots, n]$):

If $n > 1$:

Choose a pivot element p use the median

Partition around the pivot p, let $A[r]$ be the pivot

Quicksort($A[1, \dots, r-1]$)

Quicksort($A[r+1, \dots, n]$)

Alternative to Mergesort

$T(n) = 2 * T(\frac{n}{2}) + O(n)$

Quicksort

17	3	42	11	28	8	2	15	13	A

11	3	17	13	2	8	17	28	42

Arbitrage

movie is 1985

"future" is 2015

Arbitrage

Market summary > Apple Inc.
NASDAQ: AAPL - Jan 19, 7:59 PM EST

178.46 USD ↓0.80 (0.45%)
After-hours: 178.68 ↑0.12%

1 day | 5 day | 1 month | 3 month | 1 year | 5 year | max

buy low

Open	178.61	Mkt cap	916.27B
High	179.58	P/E ratio	19.43
Low	177.41	Div yield	1.41%

Arbitrage

- Input: an array $A[1, \ldots, n]$
- Output: a pair $1 \leq \boxed{i < j} \leq n$
- Goal: maximize $\text{profit}(i, j) = A[j] - A[i]$

if i, j could be anything then j [illegible]
output $i = \min$, $j = \max$
(Easy to get $O(n)$)

	i				j		
8	3	28	11	17	42	2	35
					max	min	

Naive Alg: $\Theta(n^2)$

Arbitrage: Take I

	i_L	j_L				i_R	j_R
8	3	28	11	17	42	2	35
	min				max		

Best (i,j) is the best among:

- Best (i_L, j_L) in the first half
- Best (i_R, j_R) in the second half
- Best (i_M, j_M) where i_M in first half, j_M in second half

i_M = min of first half, j_M = max of second half

Arbitrage: Take I

$T(n) = 2T(\frac{n}{2}) + O(n)$
$= \Theta(n \log n)$

8	3	28	11	17	42	2	35

MaxProfit($A[1, \dots, n]$): // Returns i, j
    If $n = 1$: return $(1,1)$ // Base Case

    $m \leftarrow \lfloor n/2 \rfloor$
    $(i^L, j^L) \leftarrow$ MaxProfit($A[1, \dots, m]$)
    $(i^R, j^R) \leftarrow$ MaxProfit($A[m+1, \dots, n]$)
// Recursively solve each half
    $i^M \leftarrow \text{argmin}\{A[1, \dots, m-1]\}$
    $j^M \leftarrow \text{argmax}\{A[m+1, \dots, n]\}$
} $O(n)$ time

    Return the best solution among L, R, M } $O(1)$ time

$A[j_L] - A[i_L]$, $A[j_R] - A[i_R]$, $A[j_M] - A[i_M]$

Arbitrage: Take I

$\min^A = \min\{\min^L, \min^R\}$ $\max^A = \max\{\max^L, \max^R\}$

	$i^L, \min^L$	$j^L, \max^L$			$\max^R$	$\min^R, i^R$	j^R
8	3	28	11	17	42	2	35

MaxProfit$(A[1,\dots,n])$: // Returns i, j, min, max

If $n = 1$: return $(1,1)$ // Base Case

$m \leftarrow \lfloor n/2 \rfloor$

$\cancel{(i^L, j^L)} \leftarrow$ MaxProfit$(A[1,\dots,m])$ $\rightarrow$ $(i^L, j^L, \min^L, \max^L)$ $i^M = \min^L$

$\cancel{(i^R, j^R)} \leftarrow$ MaxProfit$(A[m+1,\dots,n])$ $\rightarrow$ $(i^R, j^R, \min^R, \max^R)$ $j^M = \max^R$

$\cancel{i^M \leftarrow \operatorname{argmin}\{A[1,\dots,m-1]\}}$

$\cancel{j^M \leftarrow \operatorname{argmax}\{A[m+1,\dots,n]\}}$

want to reduce the $O(n)$ cost to combine.

Also have to return min, max

Return the best solution among L, R, M can do in $O(1)$ time

Speed up the algorithm by solving a slightly harder problem

Arbitrage: Take II

i^L, min^L		j^R, max^R		max^R	min^R, i^R		j^R
8	3	28	11	17	42	2	35

$p^L = A[j^L] - A[i^L] = 25$

$\boxed{p^M = A[max^R] - A[min^L] = 39}$

$p^R = A[j^R] - A[i^R] = 33$

$max = max\{42, 28\} = 42$

$min = min\{3, 2\} = 2$

return $(2, 6, 2, 42)$

i, j, min, max

Arbitrage: Take II

Momk

$T(n) = T(a) + T(b)$

$a + b \approx (1 - \frac{1}{k})n$

8	3	28	11	17	42	2	35

MaxProfitII($A[1, \dots, n]$): // Returns i, j, mi, ma,
If $n = 1$: return $(1,1,1,1)$ // Base Case

$m \leftarrow \lfloor n/2 \rfloor$
$(i^L, j^L, mi_L, ma_L) \leftarrow \text{MaxProfitII}(A[1, \dots, m])$
$(i^R, j^R, mi_R, ma_R) \leftarrow \text{MaxProfitII}(A[m+1, \dots, n])$
$i^M \leftarrow mi_L$ $\qquad j^M \leftarrow ma_R$
$mi \leftarrow \min\{mi_L, mi_R\}$ $\qquad ma \leftarrow \max\{ma_L, ma_R\}$

If L is best: return (i^L, j^L, mi, ma)
Elseif R is best: return (i^R, j^R, mi, ma)
Elseif M is best: return (i^M, j^M, mi, ma)

$T(n) = 2 \times T\left(\frac{n}{2}\right) + O(1)$

$T(n) = \Theta(n)$

$a = 2$
$b = 2$
$c = 0$

$\left(\frac{2}{2^0}\right) = 2 > 1$