Lecture 5, Tuesday 4/17/01

Order Statistics

- Read Chapter 10.
  We skip Chapter 10.1 (min/max), read at home.
  Next time we will go to Chapter 7 (Heaps).
- Problem: Find the **i-th smallest** element (**Rank-i**).
  If i=1 ⟶ Minimum
  i=n ⟶ Maximum
  i= n/2 ⟶ Median
- Possible solution:
  - Sort
  - Index into A(i).

  } $O(n \lg n)$
- We can do better !

Randomized selection

- Divide and conquer approach:

```
RS(A,p,r,i)
  if p==r  then return A(p)
  q=RandomPartition(A,p,r)
  k=q-p+1
  if i< k  then return RS(A,p,q-1,i)
     i>k   then return RS(A,q+1,r,i-k)
     i==k  then return A(q)
```

- Correctness:
  - Assume correct for size at most n=r-p+1
  - after the partition, the arrays are smaller than n, can apply induction.
  - Claim: need to search only one part
  - Explain the **3 cases**.

Performance of Random Selection

- Lucky case:

$$T(n) \le T\left(\frac{9}{10}n\right) + \Theta(n)$$

$$\left.\begin{array}{l} n^{\log_b a} = 1 \\ af(n/b) \le cf(n) \end{array}\right\} \Rightarrow \text{case } 3 \Rightarrow T(n) = O(n)$$

- What if 99/100 instead of 9/10 ??
- Bad case:

$$T(n) = T(n-1) + \Theta(n) \Rightarrow T(n) = O(n^2)$$

Analysis continued

- Let T(n) be the expected running time.
  Condition on partition outcome:

$$T(n) \le \frac{1}{n}\sum_{k=0}^{n-1} T(\max\{k, n-k-1\}) + \Theta(n)$$
$$\le \frac{2}{n}\sum_{k=n/2}^{n-1} T(k) + \Theta(n)$$

- Substitute $T(n) \le cn$, choose c large enough for T(1):

$$\left.\begin{array}{l} T(n) \le \frac{2c}{n}\sum_{k=n/2}^{n-1} k + \Theta(n) \\ \le \frac{2c}{n}\left[\sum_{k=1}^{n-1} k - \sum_{k=1}^{n/2} k\right] + \Theta(n) \\ = \frac{2c}{n}\left[\frac{n(n-1)}{2} + \frac{n/2(n/2+1)}{2}\right] + \Theta(n) \\ = cn + \underbrace{(const_1 + \Theta(n) - const_2 \cdot c \cdot n)}_{\le 0 \text{ for large enough c}} \end{array}\right\} \Rightarrow T(n) = O(n)$$

Ω

Deterministic Order Statistics

- The randomized order statistics is very fast in practice (just like quick-sort, same additional tricks will help).
- Theoretically interesting question:
  Is there a **deterministic** linear time order-statistics algorithm ?
- Deterministic selection algorithm (select ***i***-th smallest):
  - Divide **n** elements into groups of 5.
  - find **median** in each group (brute force)
  - Use select **recursively** to find **median** among n/5 medians. (i.e. select n/2-nd smallest)
  - **Partition** around this median.
  - **Recurse** on the "appropriate" part, update ***i*** if necessary.

Deterministic order statistics - cont

- Correctness - as before. All we changed was the pivot choice.
- Time:
  at least 1/2 of the medians are $\le x \Rightarrow \frac{1}{2}\left\lfloor\frac{n}{5}\right\rfloor \ge \left\lfloor\frac{n}{10}\right\rfloor$

  each median brings 3 elements $\Rightarrow 3\left\lfloor\frac{n}{10}\right\rfloor$ total

  for $n \ge 50$, we have $3\left\lfloor\frac{n}{10}\right\rfloor \ge \frac{n}{4} \Rightarrow$ at least $n/4$ elements are $\le x$.

  Similarly, at least $n/4$ elements are $\ge x$.

  $n = 10x + y,\ y \le 9$

  $3\left\lfloor\frac{n}{10}\right\rfloor = 3x,\quad \frac{n}{4} = 2.5x + \frac{y}{4} \le 2.5x + \frac{9}{4}$

  so now we solve for: $3x \ge 2.5x + \frac{9}{4},\ x$ integer

  $\Rightarrow\ x \ge 5$
- Recursion:

$$T(n) \le T(n/5) + T(3n/4) + \Theta(n)$$

Substitution: $T(n) \le cn$, choose c large enough to cover $T(1)$.

$$T(n) \le cn/5 + 3cn/4 + \Theta(n) = cn - \underbrace{(cn/20 - \Theta(n))}_{<0 \text{ for large enough c}}$$

$\Rightarrow T(n) = O(n)$

In fact, we have $T(n) = \Theta(n)$
(*Why*??)

Deterministic selection

- **Homework:**
  analyze with groups of 4 elements and groups of 6 elements.

- **Observe that we can get deterministic variant of quicksort !**
  - » Can use as a black-box O(n) partitioning into 2 equal parts.
  - » We get recurrence $T(n)=2T(n/2)+\Theta(n)$, giving us $\Theta(n \lg n)$ total running time.
  - » (do you think it will work well in practice ??)

60