CS 381

Order Statistics

The **selection problem** is the problem of computing, given a set A of n distinct numbers and a number i, $1 \leq i \leq n$, the i^{th}h **order statistics** (i.e., the i^{th} smallest number) of A.

We will consider some special cases of the order statistics problem:

- the **minimum**, i.e. the first,
- the **maximum**, i.e. the last, and
- the **median**, i.e. the "halfway point."

Order Statistics

Medians occur at $i = \lfloor (n+1)/2 \rfloor$ and $i = \lceil (n+1)/2 \rceil$. If n is odd, the median is unique, and if n is even, there are two medians.

Selection ***(Find s-th smallest element)***

Selection is a trivial problem **if the input numbers are sorted.** If we use a sorting algorithm having $O(n \lg n)$ worst-case running time, then the selection problem can be solved in in $O(n \lg n)$ time.

But using a sorting is more like using a cannon to shoot a fly since only one number needs to computed.

$O(n)$ expected-time selection using the randomized partition

Idea: In order to find the s-th order statistics in a region of size n, use the **randomized partition** to split the region into two subarrays. Let $\mathsf{k}-1$ and $n-\mathsf{k}$ be the size of the left subarray and the size of the right subarray. If $k=s$, the pivot is the key that's looked for. If $\mathsf{s} \leq \mathsf{k}-1$, look for the s-**th element in the left subarray**. Otherwise, look for the $(\mathsf{s}-\mathsf{k})$-**th one in the right subarray**

Randomized Select Pseudocode

Randomized-Select(A, p, r, s) *// return the s-th smallest element of A[p..r]*

if $(p = r)$ **then return** A[p]

$q :=$ **Randomized-Partition(A,p,r)** *// compute pivot*

$k := q - p + 1;$ *// number of elements ≤ pivot*

if $(s = k)$ **then**

return A[q] *// found i-th smallest element*

else if $(s < k)$ **then**

return **Randomized-Select**$(A, p, q-1, s)$

else

return **Randomized-Select**$(A, q+1, r, s-k)$

Analysis

Denote by

- $T(n,s)$ = expected runtime for selection of s-th statistic
- $T(n) = \max_s T(n,s)$ is the expected runtime of selection for the worst case index s

Analysis

$T(n)$ the expected runtime of selection for the worst case index s

For each i, $0 \leq i \leq n-1$, the size of the left subarray is equal to i with probability $1/n$. Assuming that the larger interval is taken, for some $\alpha > 0$, $T(n)$ is at most

Work for partition

$$\alpha n + \frac{1}{n} \sum_{1 \leq k \leq n-1, k \neq s} T(\max(k, n-k)).$$

Expected work for recursive call

This is at most

$$\alpha n + \frac{2}{n} \left(\sum_{k=\lceil n/2 \rceil}^{n-1} T(k) \right).$$

Analysis (cont'd)

Assume that there is $c > 0$ such that $T(k) \le ck$ for all $k < n$.

Then the sum $\sum_{k=\lceil n/2 \rceil}^{n-1} T(k)$ is at most $\sum_{k=\lceil n/2 \rceil}^{n-1} ck$. This is at most

$$
\begin{aligned}
& \sum_{k=1}^{n-1} ck - \sum_{k=1}^{\lceil n/2 \rceil - 1} ck \\
= \; & \frac{cn(n-1)}{2} - \frac{c}{2}\left(\left\lceil \frac{n}{2} \right\rceil - 1\right)\left\lceil \frac{n}{2} \right\rceil \\
\le \; & \frac{cn(n-1)}{2} - \frac{c}{2}\left(\frac{n}{2} - 1\right)\frac{n}{2} \\
= \; & cn\left(\frac{3n}{8} - \frac{1}{4}\right).
\end{aligned}
$$

Analysis (cont'd)

So, if c is sufficiently large,

$$T(n) \le \alpha n + c\left(\frac{3}{4}n - \frac{1}{2}\right).$$

By making the constant c at least 4α, we have that αn is at most $\frac{cn}{4}$. Then $\mathrm{T(n)} \le \mathrm{c} \cdot n$.