Randomized Select

1 Follow-up to Randomized Quicksort

In the analysis of randomized quicksort, we analyzed the probability

$$\mathbb{P}r\left[\sum_{t=1}^{T} X_{\alpha,t} \geq 2 \cdot \mathbb{E}\left[\sum_{t=1}^{T} X_{\alpha,t}\right]\right].$$

After class, many students asked why we were only concerned with when the sum is greater than a *constant multiple* of its expected value; why isn't it enough to consider when the sum is a constant number of standard deviations above the expected value? Let's examine this.

Pretend you flip n coins and store the outcomes in $y_1, \ldots, y_n$. Each y_i is 1 if coin i lands heads, and is 0 if the coin lands tails. Clearly, $\mathbb{E}y_i = 1/2$, so if we take the sum $Y = \sum_i y_i$ then we have $\mathbb{E}Y = n/2$.

Constant multiple: We can rewrite the Chernoff bound as

$$\mathbb{P}r\left[\left|\sum_i y_i - \mathbb{E}[\sum_i y_i]\right| > b \cdot \mathbb{E}[\sum_i y_i]\right] \leq 2e^{\dfrac{-b^2\mathbb{E}[\sum_i y_i]}{3}}.$$

This shows that

$$\mathbb{P}r\left[\left|Y - \frac{n}{2}\right| > b\frac{n}{2}\right] \leq 2e^{\dfrac{-b^2(n/2)}{3}} = 2e^{-cn}$$

for some constant c. Because this probability is exponential in n, we say it gives a bound "with *very* high probability."

Additive standard deviations: We want to calculate

$$\mathbb{P}r\left[\left|Y - \frac{n}{2}\right| > b\sqrt{n \log n}\right].$$

In order to calculate this, we need to rewrite $b\sqrt{n \log n} = b'(n/2)$. Solving, we need $b' = 2b\sqrt{\log n/n}$. Using the Chernoff bound:

$$\mathbb{P}r\left[\left|Y - \frac{n}{2}\right| > b\sqrt{n \log n}\right] \leq 2e^{\dfrac{-b'^2(n/2)}{3}} = 2e^{\dfrac{-4b^2(\log n/n)(n/2)}{3}} = 2e^{-(2/3)b^2 \log n} = 2n^{-c}$$

for some constant c. As discussed in class, this is a "with high probability" bound because it's polynomial in $1/n$.

Why does randomized quicksort require the very high probability bound? Remember that we had $T = 30 \log n$, so we're summing over $O(\log n)$ coin flips, not $O(n)$ flips as in the examples above. Let's see what happens if there are only $n = \log m$ flips:

Constant multiple: $2e^{-cn} = 2m^{-c}$, which is still a high probability bound.

Addititive standard deviations: $2n^{-c} = 2(\log m)^{-c}$ is no longer a high probability bound!

Thus, in order to get a high probability bound for randomized quicksort, we needed to use the constant multiple of the expected value, which is what we used in lecture.

2 Randomized Select

In this recitation we will study a randomized algorithm, RANDOMIZED-SELECT, for the k-th order statistics of an arbitrary array.

2.1 Algorithm

The algorithm RANDOMIZED-SELECT works by partitioning the array A according to RANDOMIZED-PARTITION and recurses on one of the resulting arrays.

RANDOMIZED-SELECT(A, p, r, i)

if $p = r$
  then return $A[p]$
$q \leftarrow$ RANDOMIZED-PARTITION(A, p, r)
$k \leftarrow q - p + 1$
if $i \leq k$
  then return RANDOMIZED-SELECT(A, p, q, i)
  else return RANDOMIZED-SELECT$(A, q + 1, r, i - k)$

RANDOMIZED-PARTITION(A, p, r)

$i \leftarrow$ RANDOM(p, r)
exchange $A[p] \leftrightarrow A[i]$
return PARTITION(A, p, r)

Both of the algorithms above are as in CLRS.

2.2 Analysis of Running Time

Let $T(n)$ be the expected running time Randomized Select. We would like to write out a recursion for it.

Let E_i denote the event that the random partition divides the array into two arrays of size i and $n-i$. Then we see that

$$T(n) \leq n + \sum_{i=0}^{n-1} Pr(E_i)\left(\max\left(T(i), T(n-i)\right)\right), \tag{1}$$

where by taking the $\max$ we assume that we are recursing on the larger subarray (hence we have the less than or equal sign).

For simplicity, let us assume that n is even. Note that $\max\left(T(i), T(n-i)\right)$ is always the same as $\max\left(T(n-i), T(i)\right)$. This allows us to extend the chain of inequalities to

$$T(n) \leq n + 2\sum_{i=0}^{n/2-1} Pr(E_i)\left(\max\left(T(i), T(n-i)\right)\right). \tag{2}$$

Also, since the partition element is chosen randomly, it is equally likely to partition the array into sizes $0, 1, \cdots, n-1$. So $Pr(E_i) = \frac{1}{n}$ for all i. This leads us to

$$T(n) \leq n + \frac{2}{n}\sum_{i=0}^{n/2-1} \left(\max\left(T(i), T(n-i)\right)\right). \tag{3}$$

We will not show, via substitution, that $T(n) = O(n)$.

Theorem 1 *Let $T(n)$ denote the expected running time of randomized select. Then $T(n) = O(n)$.*

Proof. We will show by the method of substitution. Let's say that $T(n) \leq cn$, and check that it works.

We must first check the base case. This is obvious, however, since $T(n')$ is a constant for some small constant n'.

Now let us check the inductive case. Assume that $T(k) \leq ck$ for all $k < n$, and we now want to show that $T(n) \leq cn$.

$$T(n) \leq n + \frac{2}{n}\sum_{i=0}^{n/2-1} \left(\max\left(T(i), T(n-i)\right)\right) \leq n + \frac{2}{n}\sum_{i=0}^{n/2-1} \left(\max\left(ci, c(n-i)\right)\right). \tag{4}$$

We note that that this is the same as

$$n + \frac{2}{n}\sum_{i=n/2}^{n-1} ci. \tag{5}$$

The term $\frac{2}{n}\sum_{i=n/2}^{n-1}(ci)$ is the same as $\frac{2c}{n}\sum_{i=n/2}^{n-1} i$. So we get

$$T(n) \leq n + c\left(\frac{2}{n}\sum_{i=n/2}^{n-1} i\right) \leq n + c\left(3n/4\right) = n\left(1 + \frac{3c}{4}\right). \tag{6}$$

Hence if we take $c = 4$ (which works for the case $T(1) \leq 4$ as well) we get

$$T(n) \leq n\left(1 + \frac{3 * 4}{4}\right) = n\,(1+3) = 4n, \tag{7}$$

as we wanted.

These notes were partly based on course notes by Avrim Blum.