CSE 241 Class 9

Jeremy Buhler

September 28, 2015

Now for randomized QuickSort!

1 Fixing Quicksort

- worst-case complexity is bad $\Theta(n^2)$
- worst case may be common (array already sorted!)
- yet it has benefits (simple, in-place)

What could we do to fix it?

- Choose some other array element consistently?
- No good, still an easy way to force $\Theta(n^2)$ performance
- Can we argue that QuickSort behaves nicely on "average" inputs? Seems hard to model "average" array
- *Better idea*: randomize the algorithm, not the inputs!

2 Randomized Algorithms

Defn: a randomized algorithm uses random numbers, *independent* of the input, in computing its answer.

- running time of algorithm depends on random choices
- can run for different times on same input
- always produces right answer (eventually)
- (such algorithms are called "Las Vegas" (vs "Monte Carlo" algos that can always finish quickly but can fail to return correct answer)
- poor performance occurs with only small probability, no matter what the input

3 Randomized Quicksort

```
QUICKSORT(A, p, r)
  if p < r
     x ← RANDOM(p, r)
     swap(A[x], A[r])
     z ← PARTITION(A, p, r)
     QUICKSORT(A, p, z − 1)
     QUICKSORT(A, z + 1, r)
```

- partitions around element chosen uniformly at random
- Let $n = r - p + 1$
- Pr(partitions around $A[j]$) $= \frac{1}{n}$, $p \leq j \leq r$

4 Analysis of Randomized Quicksort

- **Will measure expected performance**
- expectation is over all sets of random choices, *not* over inputs
- for simplicity, assume all array elements distinct
- Let $T(n)$ be expected running time of quicksort.
- **Defn**: *rank* of an element $A[x]$ is # of posns y such that $A[y] < A[x]$. (rank 0 is smallest elt)
- With probability $\frac{1}{n}$, a randomly chosen element has rank k, for each $0 \leq k \leq n-1$.
- If partition element has rank k, we partition array into parts of sizes k and $n-k-1$ (plus part elt itself).

rank	low part	high part
0	0	$n-1$
1	1	$n-2$
2	2	$n-3$
…		
$n-1$	$n-1$	0

Conclude that

$$
\begin{aligned}
T(n) &= E_k\left[\text{time with } k : n-k-1 \text{ split}\right] \\
&= E_k\left[T(k) + T(n-k-1) + cn\right] \\
&= E_k[T(k)] + E_k[T(n-k-1)] + E_k[cn] \\
&= \sum_{k=0}^{n-1} \frac{1}{n} T(k) + \sum_{k=0}^{n-1} \frac{1}{n} T(n-k-1) + cn \\
&= \frac{1}{n} \sum_{k=0}^{n-1} T(k) + \frac{1}{n} \sum_{k'=0}^{n-1} T(k') + cn \\
&= \frac{2}{n} \sum_{k=0}^{n-1} T(k) + cn
\end{aligned}
$$

5 Solving This Weird Recurrence

$$T(n) = \frac{2}{n} \sum_{k=0}^{n-1} T(k) + cn$$

How do we solve this wacky recurrence? Even a recursion tree is confusing. *Any ideas?*

- When in doubt, **guess**!
- I'm going to guess that $T(n) = \Theta(n \log n)$
- To get an answer, will need some base cases. Assume $T(0) = T(1) = 1$ (constant > 0 does not matter).
- Will show $T(n) = O(n \log n)$; Ω proof similar

Inductive proof idea:

- Will use induction on n, as usual
- **First cut at i.h.**: show that $T(n) \leq c' n \log n$ for some c' and every $n \geq 0$.
- Doesn't work in base case! Requires that $T(0), T(1) \leq 0$.
- **Second cut at i.h.**: show that $T(n) \leq c' n \log n + 1$ for some c' and every $n \geq 0$.
- Works in base case: $T(0) = T(1) = 0 + 1 = 1$.
- (Note: we could also start at some larger input size, but adding lower-order terms is a more common workaround.)

Ind: assume i.h. true for $k < n$.

$$\begin{aligned}
T(n) &= \frac{2}{n}\sum_{k=0}^{n-1} T(k) + cn \\
&\leq \frac{2}{n}\sum_{k=0}^{n-1}(c' k \log k + 1) + cn \\
&= \frac{2c'}{n}\sum_{k=0}^{n-1} k \log k + \frac{2}{n}\sum_{k=0}^{n-1} 1 + cn \\
&= \frac{2c'}{n}\sum_{k=0}^{n-1} k \log k + 2 + cn
\end{aligned}$$

We need a fancy summation formula! Can show that

$$\sum_{k=0}^{n-1} k \log k \leq \frac{1}{2}n^2 \log n - \frac{1}{8}n^2.$$

Conclude that

$$\begin{aligned}
T(n) &\leq 2 + cn + \frac{2c'}{n}\left(\frac{1}{2}n^2 \log n - \frac{1}{8}n^2\right) \\
&= 2 + cn + c' n \log n - \frac{c'}{4}n \\
&= c' n \log n + 1 + \left(cn - \frac{c'}{4}n + 1\right)
\end{aligned}$$

Need to choose c' so that, for $n \geq 2$,

$$n\left(c - \frac{c'}{4}\right) + 1 \leq 0.$$

Set $c' = 4(c+1)$, and it works! With this c', i.h. goes through. QED