Lecture 13.10.2016 COL 106

Analysing Randomised Quicksort

1. A probability problem: We toss a series of coins independently. Each ~~each~~ comes up heads with probability q and tails with prob $1-q$.

Q: What is the probability that we need at least t tosses to get k heads? Consider $t \geq 2k$ and $q \leq 1/2$

A. The probability that we have not achieved k heads in t tosses (for $t > k$) is

$$\binom{t}{k-1} q^{k-1}(1-q)^{t-(k-1)} + \binom{t}{k-2} q^{k-2} (1-q)^{t-(k-2)}$$

$$\ldots \binom{t}{1} q (1-q)^{t-1} + (1-q)^t$$

If we call this event A then

$$P(A) = \sum_{i=1}^{k} \binom{t}{k-i} q^{k-i} (1-q)^{t-(k-i)}$$

~~Since~~ If $t \geqslant 2k$, $\binom{t}{k-i} \leq \binom{t}{k-(i-1)}$ $\forall\, i : 1 \leq i-1 \leq k$

$$\Rightarrow P(A) \leq k \cdot \binom{t}{k} (1-q)^t \quad \left[\text{Since } q \leq 1/2\right]$$

Using $\binom{t}{k} \leq \left(\frac{et}{k}\right)^k$ and setting ~~k~~ $t = ck$ $(c > 1)$

we get

$$P(A) \leq k \cdot (ec)^k (1-q)^{ck}$$

$$= k \left(ec(1-q)^c \right)^k$$

Since $(1-q) < 1 \Rightarrow \exists\, c^*$ st $\forall c \geqslant c^* : c(1-q)^c < e$

[In fact c^* can be calculated easily as

~~xx~~ $\ln c + c \ln(1-q) < 1 \Rightarrow c \ln \frac{1}{1-q} > \ln c - 1$

∴ Find c^* such that $e c^* (1-q)^{c^*} = 2/3$

$$\Rightarrow \boxed{P(t \geq c^* k) \leq k \cdot \frac{2}{3^k}} \quad \text{for some constant } c^* \text{ that depends only on } q. \qquad (\dagger)$$

—— × ——

2. Pivoting an array : We are given an array A of integers. The array size is n. We are given an index i $(0 \leq i \leq n-1)$. We need to rearrange the array such that all elements less than $A[i]$ are placed to the left of $A[i]$ and all elements greater than it are placed to the right. (Note $A[i]$ itself may have to move).

Alg: Set $p \leftarrow A[i]$
Set $l \leftarrow 0$, $r \leftarrow n-1$
* While $A[l] \leq p$ { l++ }
While $A[r] > p$ { r-- }
Swap $A[l]$ and $A[r]$
Repeat from * until $l = r$

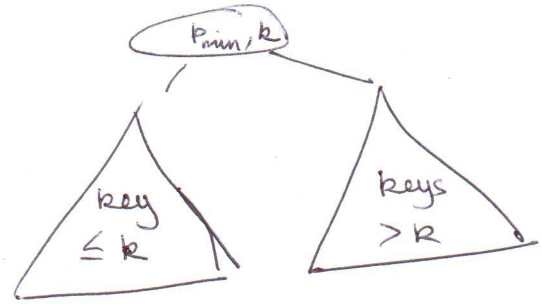
Clearly this takes $O(n)$ time.

3. Recursive pivoting with priorities : Assign distinct priorities p_i to each element in A. Now choose i such that p_i is min. Pivot the array around $A[i]$. If the new location of $A[i]$ is j then recursively do the same thing for array $A[0 \ldots j-1]$ & $A[j+1 \ldots n-1]$.
Base case: Array of size 1 is already pivoted.

This procedure sorts the array.

4. Time taken for recursive ~~pivoting~~ pivoting: Note the treap structure in the algorithm.

Left subtree has keys $\leq k$ and its root has the min priority ie a treap is formed.



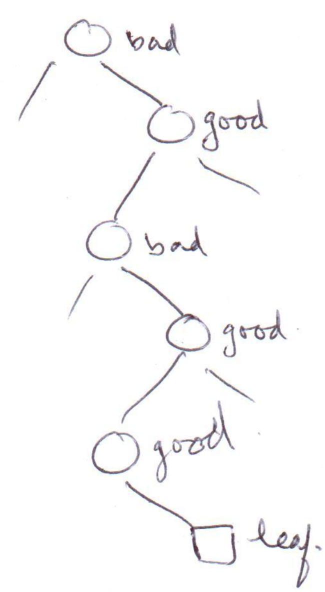
Time taken to pivot an array of size k around min priority element: $T(\text{find minpriority}) + T(\text{pivot}) = \Theta(k)$.

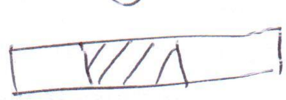
Note: Each level of the treap incurs $O(n)$ pivoting time since the sum of lengths of arrays being pivoted is $\leq n$.

$\therefore$ Time is $O(n\,h)$ where h is the height of the treap. (++)

5. Randomized treaps: Give every location a random priority selected uniformly from $[0,1]$.

Consider any leaf v of the treap. Consider the path from root to leaf. We call ~~a~~ a node in this path good if the min priority node ~~divides~~ lies in the middle third of its array (when seen in sorted order).





Probability a node is good $= \frac{1}{3}$

Every time we get a good node the largest subarray created is $2/3$ the size of the previous array.

$\therefore$ At most $\log_{3/2} n$ good nodes are possible

Noting that each node is labelled independently we get using (†) that

$$P\left[\text{length of path} \geq c^{*}.3\log_{3/2} n\right] \leq 3.\log_{3/2} n \cdot \frac{1}{n^3}$$

$$\leq \frac{1}{n^2}$$

~~$\Rightarrow$ P[length of~~ Let A_j : length of path to leaf j is $\geq c^{*}3.\log_{3/2} n$

$\therefore$ Event ht of treap $> c^{*}.3.\log_{3/2} n$

$$= \bigcup_{j=1}^{n} A_j$$

$$\Rightarrow P\left[ht > 3c^{*}\log_{3/2} n\right] = P\left[\bigcup_{j=1}^{n} A_j\right] \leq \sum_{j=1}^{n} P(A_j) \leq n.\frac{1}{n^2} = \frac{1}{n}$$

$\therefore$ Treap ht $\leq 3c^{*}\log_{3/2} n$ w prob $\geq 1 - \frac{1}{n}$

$\therefore$ Randomized quicksort takes time $3c^{*}.n\log_{3/2} n$ w. prob $\geq 1 - \frac{1}{n}$

(from (††))

$\square$