cs210 final assignment

abhinav garg

April 2016

1 Algorithm

1. Let the given input array is A.
2. Now we take a sample of size $n^{\frac{1}{d}}$ from it randomly i.e. by generating random indices.
3. Let's call the obtained array as B.
4. We now sort B.
5. then we select 2 barriers X and Y at a distance of inc indices from middle i.e. index of X is $\frac{n^{\frac{1}{d}}}{2} - inc$ and that of Y is $\frac{n^{\frac{1}{d}}}{2} + inc$.
6. we now scan A to find elements lying between X and Y and also calculate the rank of X and Y
7. If rank of X comes out to be greater than n/2 or rank of Y comes out as less than n/2 we repeat from step 2.
8. Let the new array obtained of elements between X and Y (X and Y included) be C.
9. We now sort C and report it's ($\frac{n}{2}$-rank_x+2)th element as it's median.

1.1 Names and Key's used in the following proofs

A-input array :: B-randomly selected array of size $n^{\frac{1}{d}}$ :: C-array of elements lying between X and Y :: X and Y- elements of B selected as barriers for bounding median :: t-number of times we have to repeat step 2 :: p-Number of elements of A between X and Y :: inc - increment taken from the middle index for selecting X and Y :: m-median

2 number of comparisons and analysis

First we randomly pick $n^{\frac{1}{d}}$from A. to B

Therefore expected number of keys of A lying between any two consecutive keys in B is

$$= \frac{n}{n^{\frac{1}{d}}} = n^{1-\frac{1}{d}} \tag{1}$$

Therefore, Expected number of numbers of A lying between X and Y (p) are 2*inc*$n^{1-\frac{1}{d}}$.

$$p = 2 * inc * n^{1-\frac{1}{d}} \tag{2}$$

Therefore total number of comparisons needed are $plog(p)$ once the correct barriers X and Y are found.

Number of comparisons required to sort B is $n^{\frac{1}{d}}log(n^{\frac{1}{d}})$.

Now to find number of comparisons required for computing rank of X and Y and also the numbers of A lying between them ,we need to scan A and compare each element with X if it's greater than X we compare it with Y.For,p we'll take very small value as compared to n.So,We can safely assume that n/2 numbers are smaller than X and n/2 numbers greater than Y.Therefore for n/2 numbers smaller than X we'll require 1 comparison and for other n/2 we require 2 comparison.Therefore expected number of comparisons required to compute C is 1*(n/2)+2*(n/2) = 3n/2.

Therefore ,total number of comparisons required are

$$[n^{\frac{1}{d}}log(n^{\frac{1}{d}}) + \frac{3n}{2}] * t + plog(p) \tag{3}$$

where t=number of times we calculate B

For large n,p and $n^{\frac{1}{d}}$ both decreases asymptotically . Therefore if we can show that choose d and inc such that t is 1 with high probability,we will have number of comparisons around 1.5n always.

2.1 Proof for t=1 with very high probability

For the purpose of proof choose inc such that C is of size$O(n^{\frac{1}{d}})$.Therefore let p=2*$n^{\frac{1}{d}}$.Which implies inc=$n^{\frac{2}{d}} - 1$. Now we can fail if :-

- X is greater than median m.
- Y is less than median m.

By symmetry we can argue that probability of both are same therefore,lets calculate probability of X greater than m.

2.1.1 Probability of $X > m$

For X to be greater than m .Number of elements less than m in A that are selected for B must be less than $n^{\frac{1}{d}}$/2-inc.

Let K be the number of elements in B that are less than m. Therefore we need to find P($K < n^{\frac{1}{d}}$/2-inc).This can be bounded by two methods:-

- Stirling's approximation
- chebyshev inequality.

Method using Stirling's method is very long and tedious for this report and hence we bound this number using Chebyshev inequality.

For this purpose we require E[K] and var[K]. Both of these can be easily obtained using the fact that each element has half probability that it is greater than median or by using indicator variable also these can be obtained to be as:-

$$E[K] = n^{\frac{1}{d}}/2 \tag{4}$$

$$var[K] = n^{\frac{1}{d}}/4 \tag{5}$$

Now ,

$$P(K < n^{\frac{1}{d}}/2 - inc) = P(K - E[K] < -inc) \tag{6}$$

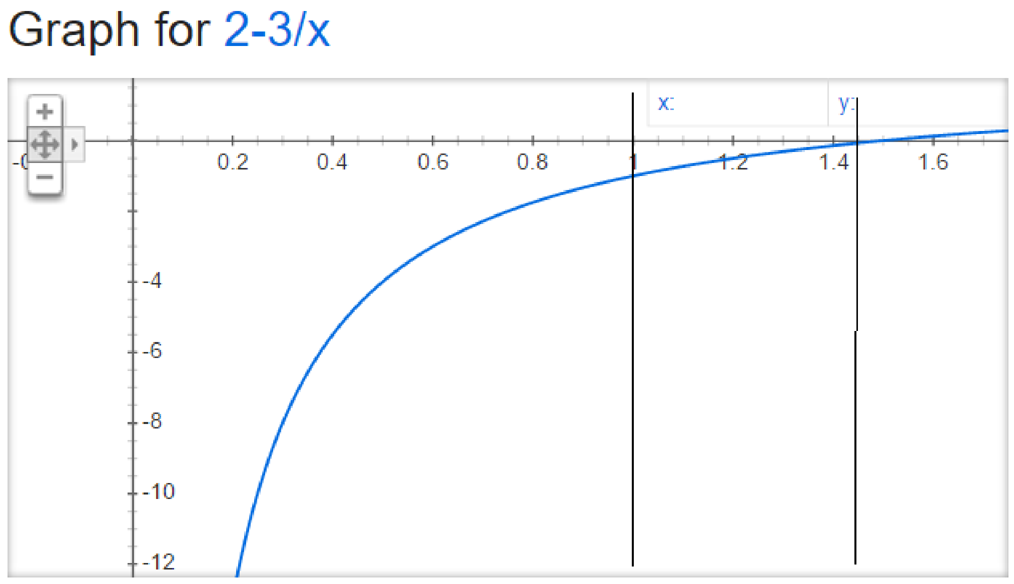
$$P(|K - E[K]| > inc) \leq \frac{var[K]}{inc^2} \tag{7}$$

$$P(|K - E[K]| > inc) \leq \frac{n^{\frac{1}{d}}}{4inc^2} \tag{8}$$

putting inc as $n^{\frac{2}{d}} - 1$

$$P(FAIL) \leq n^{2-\frac{3}{d}} \tag{9}$$

Note probability of failure is twice of equation 9 .But we can omit the constant factor for our analysis. Now we want this probability to decrease exponentially with n.And hence we want $2 - \frac{3}{d}$ to be negative.For this purpose we can plot it's graph.



We can clearly see that $d < 1.5$ and also as we know $d > 1$.And hence therefore

$$1 < d < 1.5 \tag{10}$$

also from the nature of the graph we can see that increment in y coordinate is much less than increment in x .And hence we select d as closer to 1.5 say 1.4.

3 Reason for number of TOTAL comparisons being greater than 2n in sample size less than 1 crore

It can be seen from table that total comparisons are greater than 2n for size less than 2 crore but that is not failure of algorithm that's because for this small sample size comparisons for sorting dominates that for scanning.The correctness of the algorithm can be judged by seeing the comparison excluding merging which are very close to expected $\frac{3n}{2}$ .And also comparing the expected value of failure and elements between X and Y it's apparent algorithm is a huge success even for small n.

Also we can improve our algo to work for such small numbers also by playing with the value of inc and d.A particularly good experimental results are found when inc is set to clog(n).However as we have to satisfy needs at large enough n we ignore mention of those solution in the statistics provided.

Table 1: Statistics of output for 1 lakh runs of Algorithm and d=1.4

n	no. of times,t>1such that P(fail) = P(FAIL) $\leq n^{2-\frac{3}{d}}$		no. of elements between X and Y (p) $p = 2 * inc * n^{1-\frac{1}{d}}$=2*$n^{\frac{1}{d}}$		No. of comparison excluding merge sort =$\frac{3n}{2}$		Total Comaparisons $= [n^{\frac{1}{d}} log(n^{\frac{1}{d}}) + \frac{3n}{2}] * t + 2 * n^{\frac{1}{d}} log(n^{\frac{1}{d}})$	
	Theoretical	Practical	Theoretical	Practical	Theoretical	Practical	Theoretical	Practical
1000	0.37	0	403	413	1500	1683	>2n	5601
10000	0.268	0	1421	1391	15000	15622	>2n	34363
100000	0.183	0	7428	7436	150000	153768	>2n	278521
1000000	0.139	0	38480	38273	1500000	1519327	>2n	2404043
10000000 (1 crore)	0.10	0	199114	201353	15000000	15018381	2.05n	19940306 (1.99n)
30000000 (3 crore)	0.08	0	436029	435846	45000000	45216236	1.8932n	56434556 (1.88n)
50000000 (5 crore)	0.079	0	628148	633245	75000000	75302900	1.850n	92065113 (1.84n)
70000000 (7 crore)	0.074	0	798863	799395	105000000	105403578	1.824n	125510000 (1.793n)