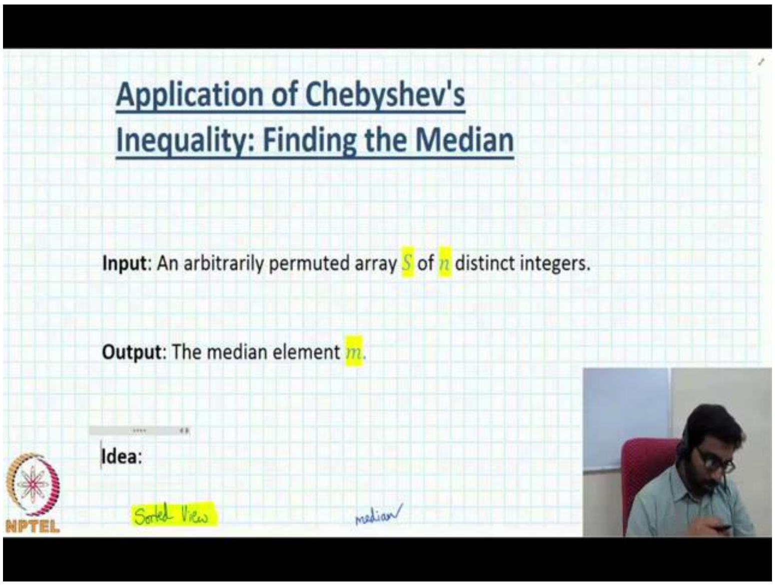
Algorithms for Big Data
Prof. John Ebenezer Augustine
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture – 13
Application of Chebyshev's Inequality

(Refer Slide Time: 00:16)



Application of Chebyshev's Inequality: Finding the Median

Input: An arbitrarily permuted array S of n distinct integers.

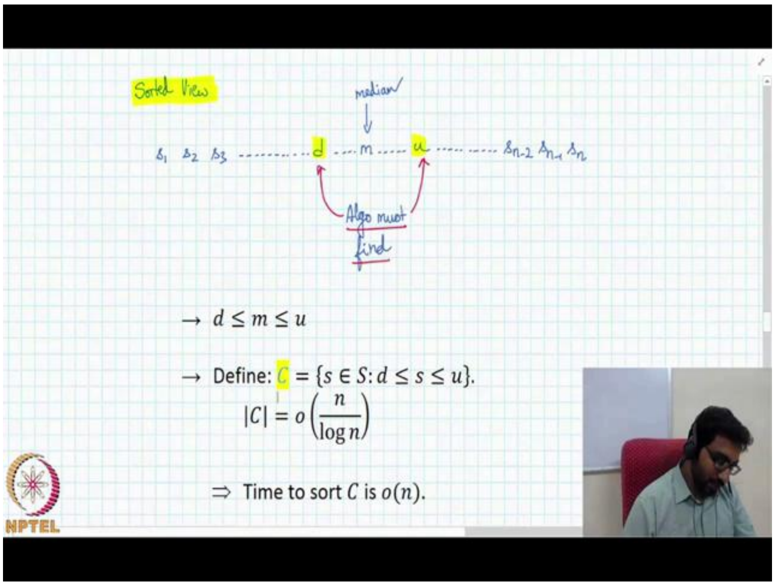
Output: The median element m.

Idea:

We are now going to look at a very fundamental problem, and it will also service an application of Chebyshev's inequality; and the problem is that of finding the median element in a num-sorted array. So, the input is an array S of n distinct integers I mean they do not really need to be distinct, but for simplicity we are going to assume that they are distinct.

And of course, we are going to assume that the arrays arbitrarily permuted because if it sorted then finding the median is trivial. So, the output the required output is the median element m, and those of you have taken basic algorithm course, metric all that there is an O of n times algorithm to find the median and that runs in deterministic O of n time is always correct. However, the advantage with of this algorithm is that it is simple to implement, and also serves to illustrate Chebyshev's inequality. So, from that point of view, we are going to study this problem.

(Refer Slide Time: 01:41)


Sorted View
median
s_1 s_2 s_3 .... d .... m ..... u ..... s_{n-2} s_{n-1} s_n
Algo must find
NPTEL

→ $d \le m \le u$

→ Define: $C = \{s \in S: d \le s \le u\}$.

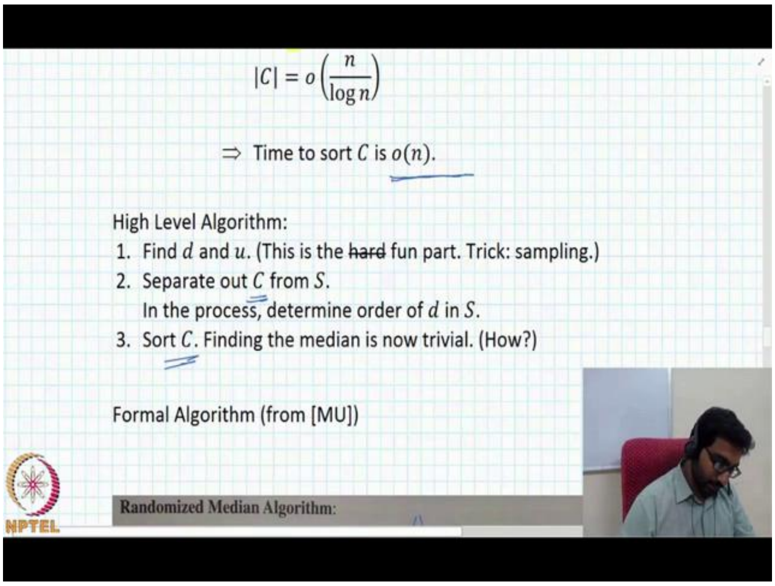
$|C| = o\left(\frac{n}{\log n}\right)$

⇒ Time to sort C is $o(n)$.

And the key idea is a following. So we have an input array S, let us view this array in its sorted form and let emphasize here that the input is arbitrarily permuted when we are talking about the sorted view, we are purely talking about the array from analysis point of view. So, there is no sorting there is actually happening. So, in this sorted view, the middle element of course, is the median; and the idea behind this algorithm is to find two elements d and u on either side of m.

And hopefully these two elements are close to m. So, what do we mean by that. Let us define C to be the set of all elements in the array S that lie between the end u. And we want that set C to be small in particular we want that to be little o of n over log n. And this would mean that if you sort C, you only require this o of n time. And if you are able to find such elements d and u and such that these properties hold, so d and u are either side of m and the set C that we have defined here are small, these are the requirements. Now, suppose we are able to find these elements then we can find the median and here is how that works.

(Refer Slide Time: 03:35)



$$|C| = o\left(\frac{n}{\log n}\right)$$

$\Rightarrow$ Time to sort C is $o(n)$.

High Level Algorithm:

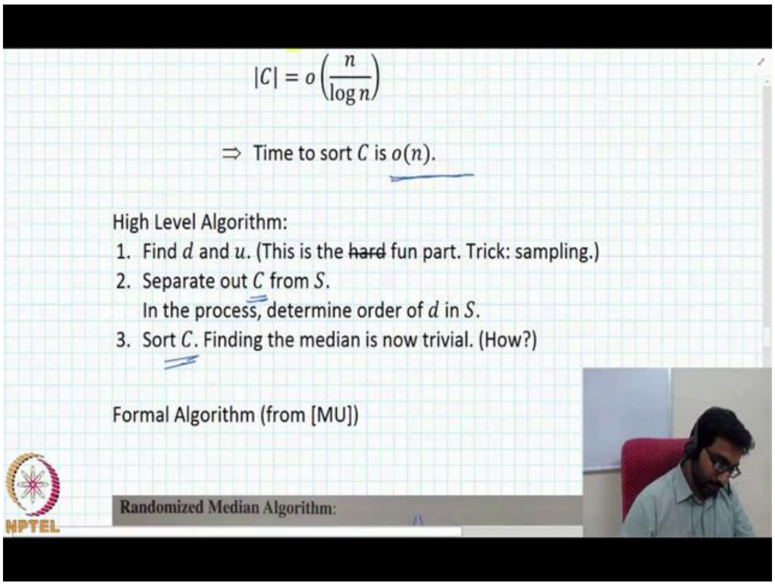
1. Find d and u. (This is the ~~hard~~ fun part. Trick: sampling.)
2. Separate out C from S.
   In the process, determine order of d in S.
3. Sort C. Finding the median is now trivial. (How?)

Formal Algorithm (from [MU])

Randomized Median Algorithm:

So, we are able to find the d and u as we required then it is easy for us to compute C, you just have to scan the entire set S, the array S and find all the elements that have values between d and q and so that will provide us with the set C. In that process, you can also find the order of dth, the position of d in the sorted view of S, think about that that is fairly easy to find. And so then what we do is we sort the set C, and I think about it, it is given the order over the position of the item d in the sorted view of S and the faculty of sorted C, it should be easy now to find the median. So, take a moment, pause and make sure you know how the median can be computed under this context.

(Refer Slide Time: 05:00)



$$|C| = o\left(\frac{n}{\log n}\right)$$

$\Rightarrow$ Time to sort C is $o(n)$.

High Level Algorithm:

1. Find d and u. (This is the ~~hard~~ fun part. Trick: sampling.)
2. Separate out C from S.
   In the process, determine order of d in S.
3. Sort C. Finding the median is now trivial. (How?)

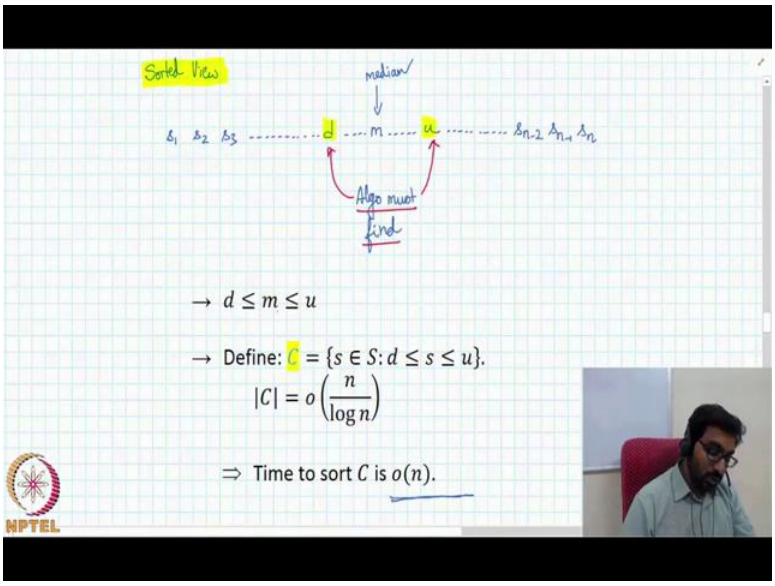
Formal Algorithm (from [MU])

Randomized Median Algorithm:

Now, that if I had a chance to think a little bit and make sure you understand the high level idea, let us actually look at the details. So, here is how the formal algorithm goes, this algorithm is taken straight out of (Refer Time: 05:08). So, the notations are as we have defined the set S is the set of elements that we want to order from which we want to find the median. And here is how the algorithm goes and as first we need to figure out how to compute d and u, and how do we do that we achieve that by sampling. So, we have to pick a set R, where each element in this R is chosen uniformly at random so that R for us, each element is in R is chosen uniformly at random and independent of each other from the set S.

And how many such elements do we choose, we choose n to the three-fourth. And then we sort that set R questions is n to the three-fourth. We should be able to do that sorting in little o of n time. And then we identify elements d and u in the following manner. Notice that since there are n to the three-fourth elements in R, n to the three-fourth elements divided by 2 will be the middle elements if you were to sort R. So, from that middle element, you walk, square root of n steps to the left you will get d. You walk square root of n steps to the right, you will get u; these are your d and u and these are elements chosen from R, but they are actually originally elements are from s.

(Refer Slide Time: 07:28)


Sorted View
median
s_1 s_2 s_3 .... d .... m ..... u .... s_{n-2} s_{n-1} s_n
Algo must find
NPTEL

→ $d \leq m \leq u$

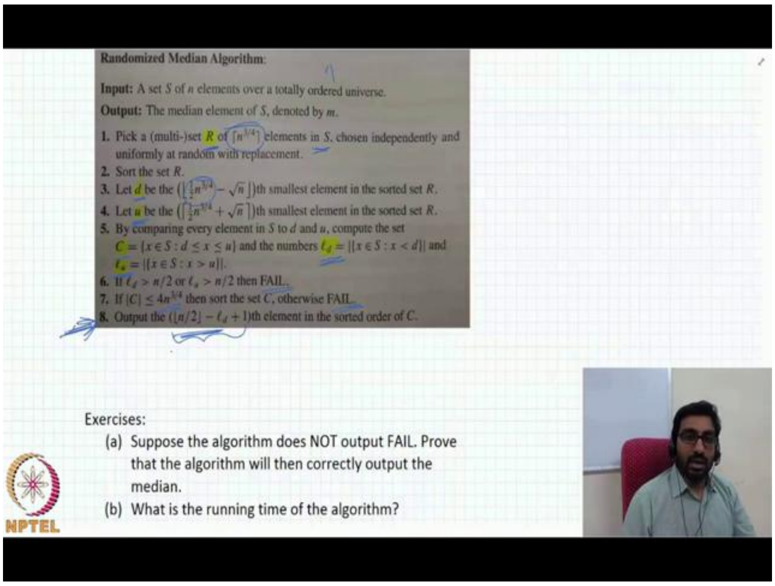
→ Define: $C = \{s \in S : d \leq s \leq u\}$.

$$|C| = o\left(\frac{n}{\log n}\right)$$

⇒ Time to sort C is $o(n)$.

Now, that we have identified the elements d and u, we have to conform that these are actually good elements, because where d and u have to follow some rules. If you recall d and u have to be on either side of m in the sorted view of s and the set of set C that we define here a set of elements in s that are between d and u should be small that cardinality of that set must be small.

(Refer Slide Time: 07:50)



Randomized Median Algorithm:

Input: A set S of n elements over a totally ordered universe.

Output: The median element of S, denoted by m.

1. Pick a (multi-)set R of $\lceil n^{3/4} \rceil$ elements in S, chosen independently and uniformly at random with replacement.
2. Sort the set R.
3. Let d be the $(\lfloor \frac{1}{2} n^{3/4} - \sqrt{n} \rfloor)$th smallest element in the sorted set R.
4. Let u be the $(\lceil \frac{1}{2} n^{3/4} + \sqrt{n} \rceil)$th smallest element in the sorted set R.
5. By comparing every element in S to d and u, compute the set $C = \{x \in S : d \leq x \leq u\}$ and the numbers $\ell_d = |\{x \in S : x < d\}|$ and $\ell_u = |\{x \in S : x > u\}|$.
6. If $\ell_d > n/2$ or $\ell_u > n/2$ then FAIL.
7. If $|C| \leq 4n^{3/4}$ then sort the set C, otherwise FAIL.
8. Output the $(\lfloor n/2 \rfloor - \ell_d + 1)$th element in the sorted order of C.

Exercises:

(a) Suppose the algorithm does NOT output FAIL. Prove that the algorithm will then correctly output the median.

(b) What is the running time of the algorithm?

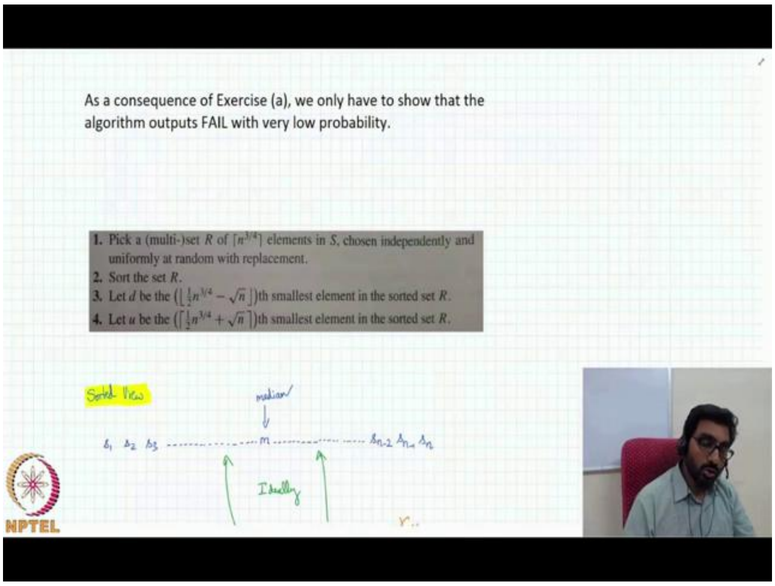
So, we have to verify that we actually have good pair d and u. How do we do that well we compute the set C straight out of the definition, but this will require an o of n scan through the set of elements in x. And then, we find the number of elements that are elements there are less than d and the number of elements that are greater than u. Now we need to make sure that the there are a few conditions are satisfied. The number of elements less than d should not exceed n by 2; if that was a case when the median is smaller than d and so that is a bad choice of d.

Similarly, the number of elements that are larger than u should also the less than n by 2 if it is greater than n by 2 then again your choice of u has been bad. And finally, the set C its cardinality has to be small; in particular we (Refer Time: 09:10) there has to be no more than 4 times n to the three-fourths, if it is more than that then again the out choice of d and u is bad. So, we want to verify that our choice of d and u are good. If we realise that the choice of d and u are bad, we immediately output a fail.

Now that we ensure now where a step 8 here we will ensure that it is not a failure our choice of d and u are both, so that at this point we can easily output the median. And you should be able to figure out why this particular element in this sorted ordering of C is the correct median. So, here are the couple of exercises for you.

Suppose the algorithm does not output fail meaning it is come up to line number 8 here. Then prove that the algorithm will in fact correctly output the median that simply requires you may ensure that this particular element that we are outputting is in fact, the correct median so that is the first exercise for you, just ensure that you convince this correct. And also convince yourself that the running time of this algorithm is at most O of n.

(Refer Slide Time: 10:55)



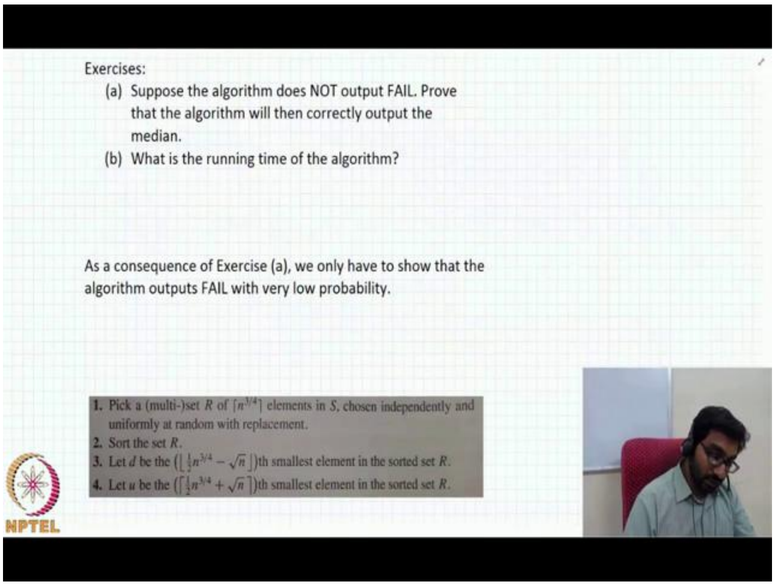
As a consequence of Exercise (a), we only have to show that the algorithm outputs FAIL with very low probability.

1. Pick a (multi-)set R of $\lceil n^{3/4} \rceil$ elements in S, chosen independently and uniformly at random with replacement.
2. Sort the set R.
3. Let d be the $(\lfloor \frac{1}{2}n^{3/4} - \sqrt{n} \rfloor)$th smallest element in the sorted set R.
4. Let u be the $(\lceil \frac{1}{2}n^{3/4} + \sqrt{n} \rceil)$th smallest element in the sorted set R.

Sorted View
median
s_1 s_2 s_3 .... m .... s_{n-2} s_{n-1} s_n
Ideally
r..
NPTEL

Now that if convinced yourself of those two claims, now let us move onto the important aspect of bounding the probability of failure that is the only important thing, key thing that is left you want to ensure as a consequence of exercise a.

(Refer Slide Time: 11:09)



Exercises:

(a) Suppose the algorithm does NOT output FAIL. Prove that the algorithm will then correctly output the median.

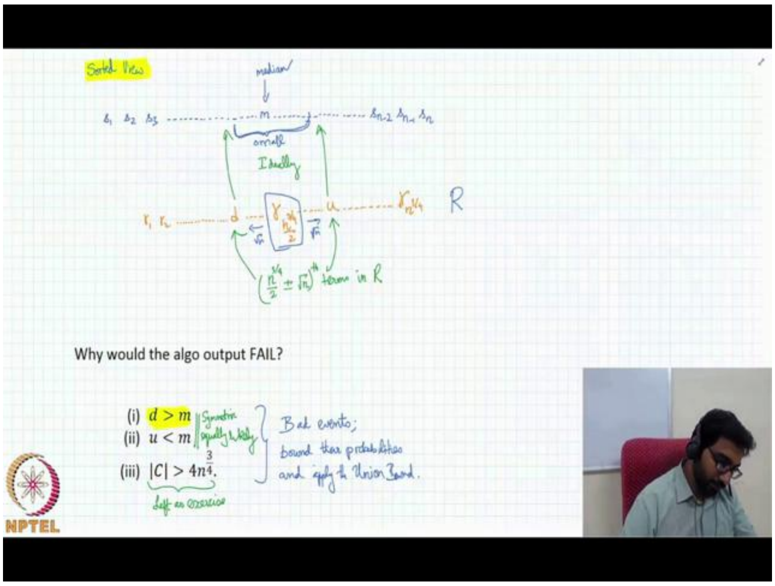
(b) What is the running time of the algorithm?

As a consequence of Exercise (a), we only have to show that the algorithm outputs FAIL with very low probability.

1. Pick a (multi-)set R of $\lceil n^{3/4} \rceil$ elements in S, chosen independently and uniformly at random with replacement.
2. Sort the set R.
3. Let d be the $(\lfloor \frac{1}{2}n^{3/4} - \sqrt{n} \rfloor)$th smallest element in the sorted set R.
4. Let u be the $(\lceil \frac{1}{2}n^{3/4} + \sqrt{n} \rceil)$th smallest element in the sorted set R.

You know if it does not output fail, you are correct. So, you just to make sure that the probability of failure are very low.

(Refer Slide Time: 11:25)



Sorted View
median
s_1 s_2 s_3 m s_{n-2} s_{n-1} s_n
overall
Ideally
r_1 r_2 d $r_{\frac{n^{3/4}}{2}}$ u $r_{n^{3/4}}$ R
$\sqrt{n}$
$\sqrt{n}$
$\left(\frac{n^{3/4}}{2} \pm \sqrt{n}\right)^{th}$ term in R

Why would the algo output FAIL?

(i) $d > m$ Symmetric
(ii) $u < m$ equally likely
(iii) $|C| > 4n^{\frac{3}{4}}$. left as exercise

Bad events; bound their probabilities and apply the Union Bound.

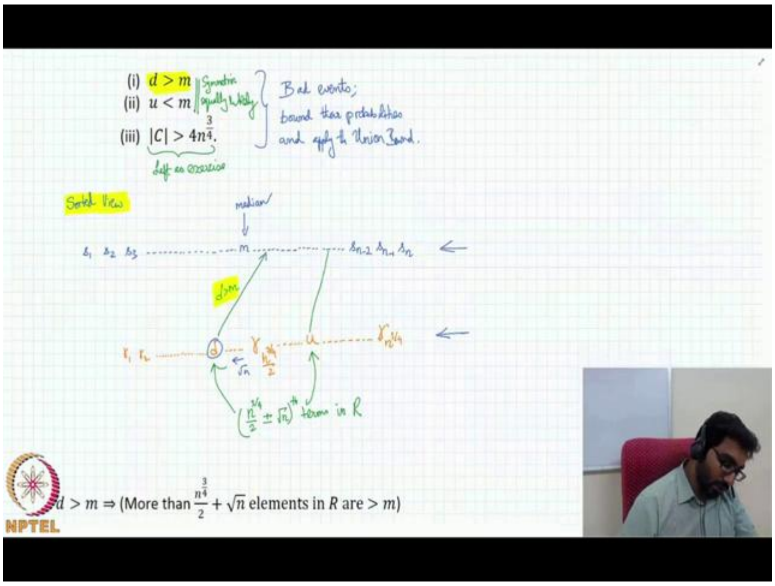
So, let us look at it bit more carefully. Feel free to pause, and start at this figure to convince yourself that the claims you are making correct. (Refer Time: 11:36) is available here for you as very reference, see what happens. So, we have the set S and we have shown in the set S in its sorted view. Now of course, this is only for analysis purpose, and the median m is the middle over there and we are sampling our multi set R which is this 1 over here, this is the multi set R, and we are actually sorting R. So, once we sort R, it is going to be appearing as R 1, R 2 and so on up to R n to the three-fourth, why because the set R has cardinality n to the 3 fourth. This set R itself has median and that is this elements over here.

And from that median element, we walk square root of n steps to the left, and look at the elements over there that is our designated d; from the median element, we walk square root of n steps to the right and that will be our designated u. And ideally, we want d to fall, if you look at d not in the set R, but in the sorted view of the input array S, d should be to the left of m, and u should be to the right of m that is the ideal requirement. And more over you also want the number of items here to be small. So, these are the ideal requirement that we have.

Now, since our main focus is to, bound the probability of failure; let us try to understand why this algorithm might fail, what are some events that will lead to the failure of this algorithm. The 3 events that we will lead to failure - the first where really symmetrically equivalent of each other, so let us look at the first manner in which the algorithm can fail that the element d that we choose according to the algorithm turns out to be larger than m meaning in the sorted view of S, d appears to the right of m then that would be a bad event, because now what we wanted was d to be to the left of m, u to be to the right of m.

Similarly, the second events which is capturing the mirror image where the bad events here is that u is appearing is less than m which means it is appearing to the left of m in the sorted view of S, so both of those are bad. And of course, the third bad event is the set C, the elements between S between d and u has cardinality that is more than 4 n to the three-fourths. So, these are the 3 bad events under which the algorithm is going to output fail, and we want to ensure that these three bad events occur individually with very low probability, and then we simply apply the union bound to say that even collectively they have low probability.

(Refer Slide Time: 15:44)



(i) $d > m$

(ii) $u < m$

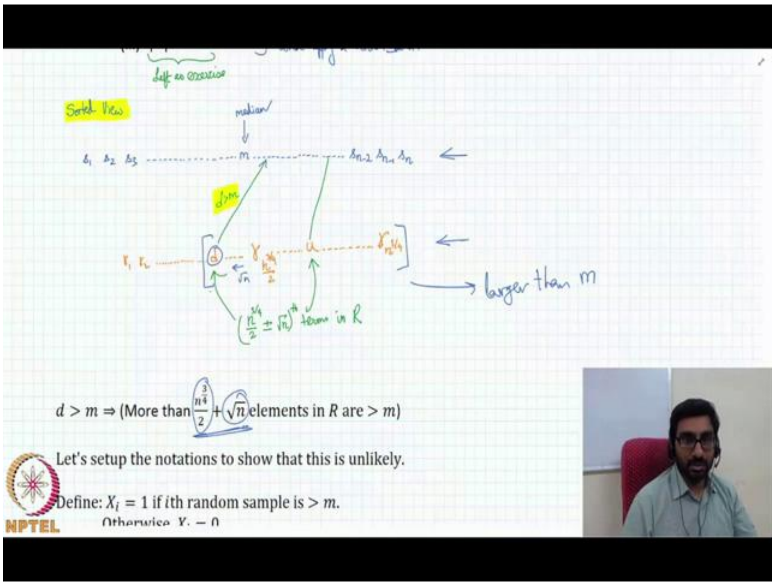
(iii) $|C| > 4n^{\frac{3}{4}}$.

$d > m \Rightarrow$ (More than $\frac{n^{\frac{3}{4}}}{2} + \sqrt{n}$ elements in R are $> m$)

So, in the interest of keeping this lecture short, the third event, the task of bounding the probability of the third bad event the cardinality of C greater than 4 into the three-fourths is left as an exercise. As I pointed out earlier, even one and two are symmetric equivalence of each other. So, we are only going to focus on the first bad event, so we want to show that the first bad event namely the d greater than m happens with low very probability.

So, let us look at the given a bit more carefully. So, when in this so recall that we have the sorted view of S over here, and this is the sample set R, and we take the median in the set R walk a square root of n steps to the left we find the d and unfortunately d happens to be larger than m. So, this is the way in which this bad event occurs.

(Refer Slide Time: 17:07)


left as exercise
Sorted View
median
s_1 s_2 s_3 ... m ... s_{n-2} s_{n-1} s_n
d>m
r_1 r_2 ... d ... $r_{\frac{n^{3/4}}{2}}$ ... u ... $r_{n^{3/4}}$
$\sqrt{n}$
$\left(\frac{n^{3/4}}{2} \pm \sqrt{n}\right)^{th}$ term in R
larger than m

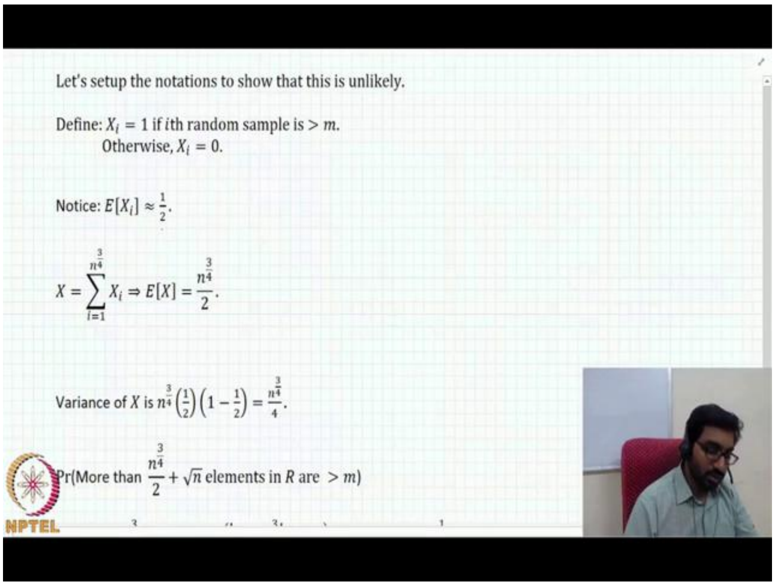
$d > m \Rightarrow$ (More than $\frac{n^{\frac{3}{4}}}{2} + \sqrt{n}$ elements in R are $> m$)

Let's setup the notations to show that this is unlikely.

Define: $X_i = 1$ if ith random sample is $> m$.
Otherwise, $X_i = 0$

So, how can this what is this mean, this means that more than n to the three halves a three quarters divided by 2 plus square root of n elements in R are larger than m. What we mean by that, well, all of these events I mean all of these items are all larger than m. And what is the cardinality of the set that is this, remember this is the median element and we walked square root of n steps to the left so that we have to add that this 1. And obviously, this looks like an unlikely event, but now we have to formally proof that it is indeed unlikely. So, let us set up some notation to do that.

(Refer Slide Time: 18:11)



Let's setup the notations to show that this is unlikely.

Define: $X_i = 1$ if ith random sample is $> m$.
Otherwise, $X_i = 0$.

Notice: $E[X_i] \approx \frac{1}{2}$.

$$X = \sum_{i=1}^{n^{\frac{3}{4}}} X_i \Rightarrow E[X] = \frac{n^{\frac{3}{4}}}{2}.$$

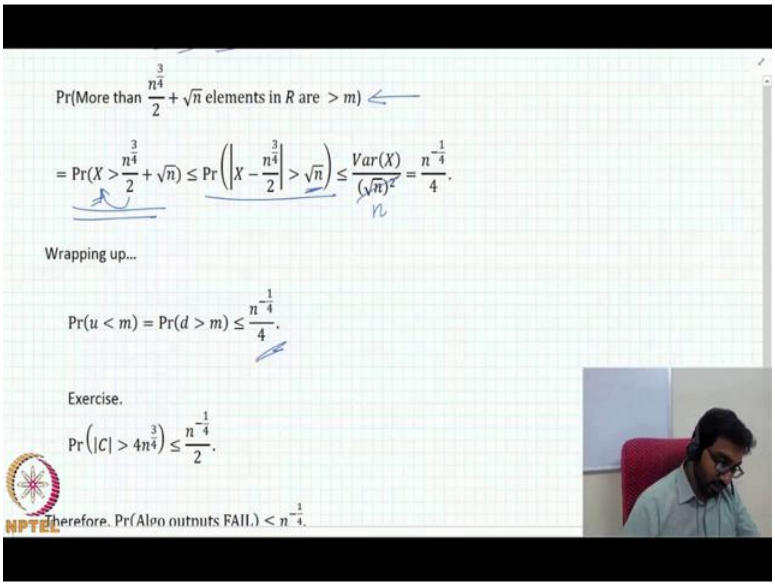
Variance of X is $n^{\frac{3}{4}}\left(\frac{1}{2}\right)\left(1 - \frac{1}{2}\right) = \frac{n^{\frac{3}{4}}}{4}$.

Pr(More than $\frac{n^{\frac{3}{4}}}{2} + \sqrt{n}$ elements in R are $> m$)

So, what we are going to do is define a Bernoulli variable X i will be a 1 is the ith random sample, you know the set R is obtained by sampling from the set S. So, the X i equal to 1, if the ith random sample is greater than n well that is remember that is the bad event where when you have lots of items in particular more than this many items larger than m that is the bad event. So, with that in mind, we are setting up the X i(s) in this manner. So, now, that you have defined X i to be 1, if the ith random sample is greater than n otherwise X i is equal to 0. So, this is our Bernoulli random variable.

And it is the expectation of x i is close to a half, it is very, very close to a half. So, we were just going to assume that it is a half. And of course, now with these we aggregate these indicator random variables to define the set x to be the sum of all these indicator random variable, and there is a total of n to the three-fourth such indicator random variable, because R has cardinality n to the three-fourths. So, the expectation on x is there it will be n to the three-fourths divided by 2, so that is this. Notice that x is a binomial random variable, so its variance is actually given by the number of Bernoulli random variables times the probability of the success of each Bernoulli random variable times the probability of failure of each Bernoulli random variable. So, the variance of x turns out to be n raise to the three-fourths divided by 4.

(Refer Slide Time: 19:57)



Pr(More than $\frac{n^{\frac{3}{4}}}{2} + \sqrt{n}$ elements in R are $> m$)

$$= \underline{\Pr\left(X > \frac{n^{\frac{3}{4}}}{2} + \sqrt{n}\right)} \leq \underline{\Pr\left(\left|X - \frac{n^{\frac{3}{4}}}{2}\right| > \sqrt{n}\right)} \leq \frac{Var(X)}{\underset{n}{(\sqrt{n})^2}} = \frac{n^{-\frac{1}{4}}}{4}.$$

Wrapping up...

$$\Pr(u < m) = \Pr(d > m) \leq \frac{n^{-\frac{1}{4}}}{4}.$$

Exercise.

$$\Pr\left(|C| > 4n^{\frac{3}{4}}\right) \leq \frac{n^{-\frac{1}{4}}}{2}.$$

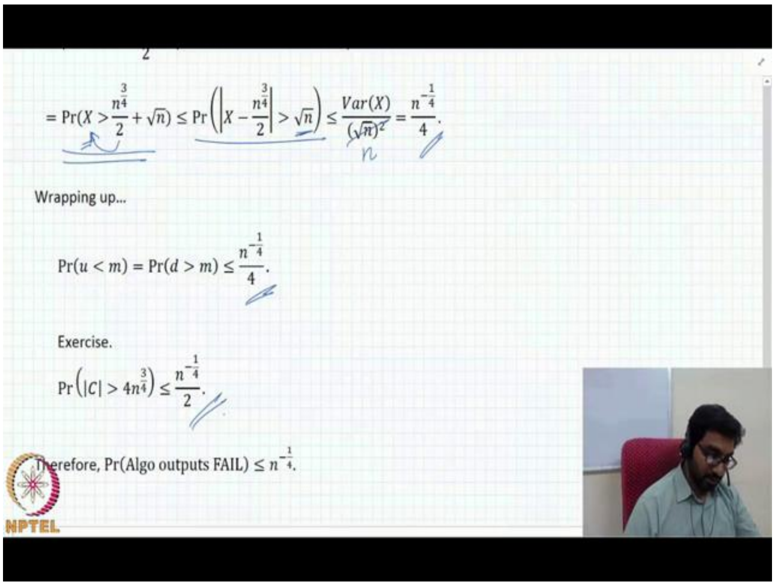
Therefore, Pr(Algo outputs FAIL) $< n^{-\frac{1}{4}}$.

So, now let us get reminder servers what the bat event is we are trying to shows unlikely, we were trying to show that the bad event of more than n to the three-fourths divided by

2 plus square root of elements in R or greater than m is small, so that is this probability that we have over here. And that probability can be phrased using the notation that we have Del, so that probability is equal to the probability that the random variable x is greater than n to the three-fourths by 2 plus square root of n. And let us do little rearrangement, we bring the n, this term over to this side and within interest in applying Chebyshev's inequality, we take the absolute value. So, we are actually slightly over counting here, the probability that the absolute value of the difference between x and n to the three-fourths divided by 2 is greater than square root of n. This event as greater probability than the probability of event that we are actually interest in that is good we want an upper bound on the probability of bad event.

And now it us in a form that we with which we can apply Chebyshev's inequality and this probability is at most variance of the random variable x divided by the square of the right hand side over here, which is and this terms is square root of n, this is essentially is just divided by n. And so this whole probability turns out to be n to the minus 1 by 4 divided by 4. So, by symmetric, the probability that u is less than m is also at most n to the minus 1 by 4 divided by 4.

(Refer Slide Time: 23:02)



$$= \Pr(X > \frac{n^{\frac{3}{4}}}{2} + \sqrt{n}) \le \Pr\left(\left|X - \frac{n^{\frac{3}{4}}}{2}\right| > \sqrt{n}\right) \le \frac{Var(X)}{\underset{n}{(\sqrt{n})^2}} = \frac{n^{-\frac{1}{4}}}{4}.$$

Wrapping up...

$$\Pr(u < m) = \Pr(d > m) \le \frac{n^{-\frac{1}{4}}}{4}.$$

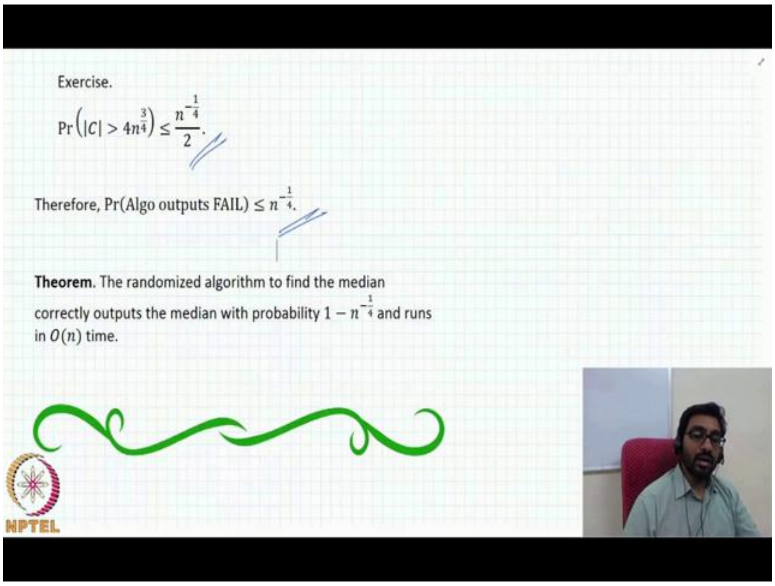
Exercise.

$$\Pr\left(|C| > 4n^{\frac{3}{4}}\right) \le \frac{n^{-\frac{1}{4}}}{2}.$$

Therefore, $\Pr(\text{Algo outputs FAIL}) \le n^{-\frac{1}{4}}$.

As an exercise, you will have to work out that the probability that the cardinality of the set C is greater than 4 n to the three-fourths is also small, and particular it is no more than n to the minus 1 by 4 divided by 2. Now, these are all the probabilities of the 3 bad events.

(Refer Slide Time: 23:33)



Exercise.

$\Pr\left(|C| > 4n^{\frac{3}{4}}\right) \leq \frac{n^{-\frac{1}{4}}}{2}$.

Therefore, Pr(Algo outputs FAIL) $\leq n^{-\frac{1}{4}}$.

Theorem. The randomized algorithm to find the median correctly outputs the median with probability $1 - n^{-\frac{1}{4}}$ and runs in $O(n)$ time.

And when you add those 3 probabilities, essentially applying the union bound, you can infer that the probability that the algorithm will output of fail is at most n to the minus 1 by 4. So, with that we can conclude with our theorem, which states that the randomised algorithm that we have studied for finding the median of set of n integers correctly outputs the median with probability 1 minus n to the minus 1 by 4 and runs in O of n time.

With that, we conclude the study of R median algorithm.