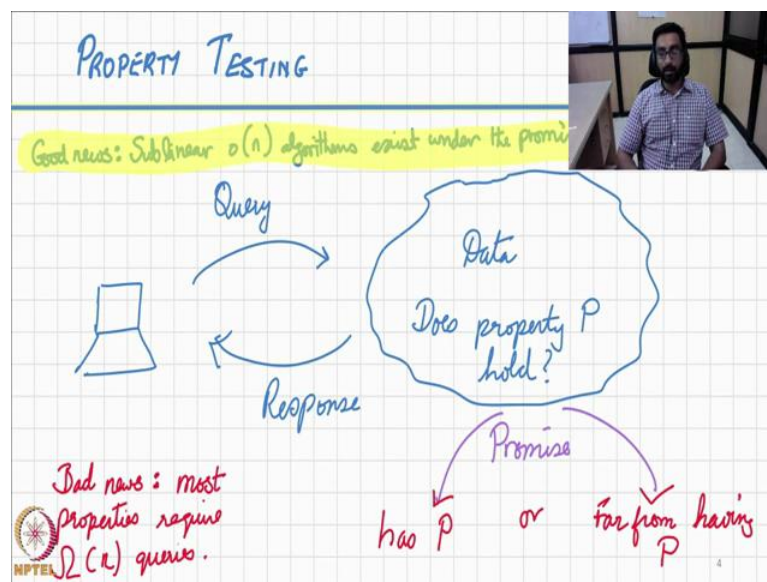


**Algorithms for Big Data**  
**Prof. John Ebenezer Augustine**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Madras**

**Lecture - 39**  
**Property Testing: The Enforce and Test Technique**

Hello everybody. We are going to continue with the topic of property testing and in particular, we are going to look at new technique called the enforce and test technique, but before we get into that technique, let us briefly remind our source about the property testing context.

(Refer Slide Time: 00:35)



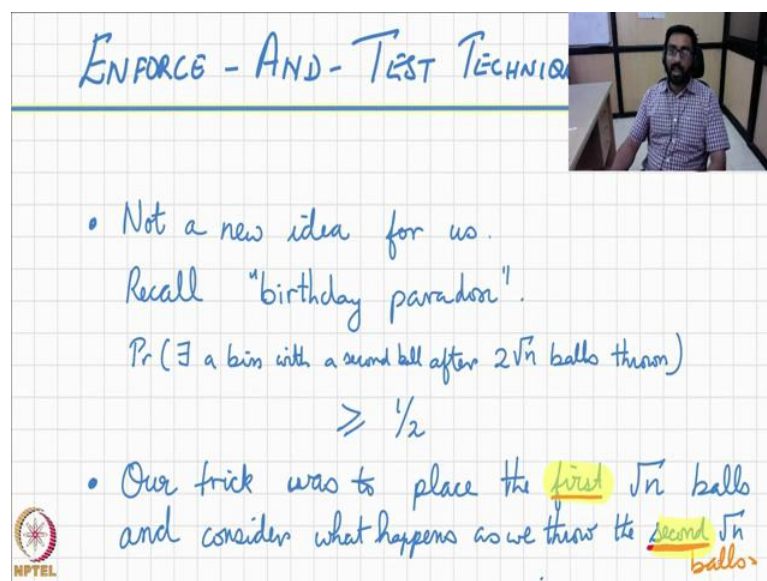
The motivation of this competition model comes from the fact that much of today's data is actually stored in large data centers and when we want to perform some competition on this data, we do not have immediate access to the data as in the traditional setting where the data is residing inside the memory of the computer which the competition is executing rather the data is available only through some interface, and we can access the data only via queries and get the response to these queries and then, perform the competition based on just our knowledge of the data through these queries and responses.

Unfortunately, many of the types of competition that we are interested in might require a lot of information. So, for example, if we wanted to find out whether a graph is connected, we only know few ways to do it and all of them require to know the entire graph, but on the other hand we had some very good success and testing whether graph is connected by making a few simple assumptions that are quite reasonable in many applications today and the assumption is that the data comes to the promise at either has the property that we are interested in or it is far from having the property that we are interested in.

If you are wondering why this is even useful in today's context, just think about situations like recognizing someone's face. Either the photograph is that person or it someone completely different and therefore, completely far away from that person and in many such applications are bound. So, we are not going to worry too much about, but we are going to embrace this promise type of problem knowing that it is realistic in many context and b, it opens up some algorithmic possibilities that do not exist in traditional algorithm in traditional competition.


So, that is the good news that motivates this study of sub linear algorithms that focus on algorithm that run with just little low of n number of queries into the data set.

(Refer Slide Time: 03:32)



**ENFORCE - AND - TEST TECHNIQUE**

- Not a new idea for us.  
Recall "birthday paradox".  
 $\Pr(\exists \text{ a bin with a second ball after } 2\sqrt{n} \text{ balls thrown}) \geq \frac{1}{2}$
- Our trick was to place the **first**  $\sqrt{n}$  balls and consider what happens as we throw the **second**  $\sqrt{n}$  balls.



This technique that we going to talk about enforce and test technique is not really new to us. We are encountered it in some slight way. Little while ago all you have to read, do a recall the birthday paradox, where in the balls and bins setting and we wanted to prove that when about two square root of  $n$  balls are thrown each uniformly at random into  $n$  bins, then with some good probability we are likely to see some bin having more than one ball on it and in order to prove this, there are some several other ways to do it, but there was one very important way in which approach, the analysis of this setting what we did was we first through square root of  $n$  balls and we just assumed there was no pairing of balls, no bin with more than one ball on it.

Well, if we did end up having a second ball, then the theorem is only strengthened. So, we place the first square root of  $n$  balls and this imposes a structure that we can then exploit. So, the first square root of  $n$  balls means that when they are through into and bins means that square root of  $n$  bins are non-empty and every subsequent balls that is thrown has  $1$  over square root of  $n$  chance of following into one such bin and that is what we exploited and this is. So, you if you just think about it, the first square root of  $n$  balls, just enforce certain structure and the second square root of  $n$  balls exploit that structure and this is the general paradigm of this enforce and test technique, but we need to be bit careful about this.

(Refer Slide Time: 05:38)

The slide is titled "ENFORCE - AND - TEST TECHNIQUE" in blue handwriting. It features a diagram on a grid background. A blue rectangle represents the "Universe of legal inputs". Inside it is a red circle labeled  $I$ . A pink arrow points from the circle to the text "Set of inputs far from having property P.". Another pink arrow points from the circle to the equation  $Pr(I \text{ is accepted}) \leq P$ . A third pink arrow points from the circle to the question "How to prove that no instance  $I \in I$  is accepted?". Below this, the words "UNION SOUND" are written in pink and crossed out with a red 'X'. Underneath that, "Too WEAK" is written in pink. To the right, the phrase "Now what?" is written in yellow on a yellow highlight. In the top right corner, there is a small video inset showing a man with glasses and a beard. In the bottom left corner, there is a logo for NPTEL.

So, let us think through this carefully now. Let us say this rectangle here represents the universe of all legal inputs and within that we have a set  $i$  and this is the set  $i$  of inputs. There are far from having the property that we are interested in testing. So, when we get an instance from this set  $i$ , we must reject that instance with the certain amount of probability or in other words, we must only accept it with some small probability that at some  $p$  and let us say we do that.

Now, we need to extend that to prove that no instance  $i$  is accepted with any reasonable probability. Well what are the ways in which we can do that? We can use the union bounds. So, we can say if this one instance is not accepted with probability  $p$ , then all of these instances are also not accepted with some probability that is equal to the cardinality of the set  $i$  times  $p$ .

So, that is too weak because if  $p$  is for this case, I mean the set  $i$  is going to be quite large typically exponentially large. The probability  $p$  has to be exponentially small in order for this union bound technique to work. So, that is usually not the case. So, this approach is too weak. This is just a sort of motivation for why we need to enforce and test technique because now we ask what we do what we seem to be stuck.

(Refer Slide Time: 07:50)

The slide is titled "ENFORCE - AND - TEST TECHNIQUE" in blue handwriting. It features a video inset in the top right corner showing a man with a beard and glasses speaking. The main content is handwritten in blue and green ink. A bullet point states "Either in algorithm or analysis". Below this, "ENFORCE" is written in blue with a pink arrow pointing to the right, followed by the text "First few queues enforce some structure into instances that must be accepted" in pink. Underneath "ENFORCE" is "AND TEST" in green, with "TEST" underlined. A green arrow points down from "TEST" to the text "Search for violation of the structure and reject" in green. In the bottom left corner, there is a logo for NPTEL (National Programme on Technology Enhanced Learning) and the number "12" in the bottom right corner.

Let us think about this. What we do the way we get round is, as we get in the balls and bins case, first try to enforce some structure and it can either be done in the context of the algorithm itself. So, if the algorithm does it, we can think of it as a sort of an explicit enforcing of the structure or it will be done in analysis where (Refer Time: 08:17) bit more implicit.

First few queries are enforced the structure and it typically enforces the structures into instances that must be accepted and then, we test against that enforced structure. In other words, we are searching for violations because once the structure is established, then it becomes easier to test and find out ways in which that structure gets violated and then, if it gets violated where we know the answer, we have to reject that instance, but if it not violated, then we accept that. This is the general approach we take in enforce and test technique.

With that, we will now look at a concrete example where this enforced and test technique is exploited.