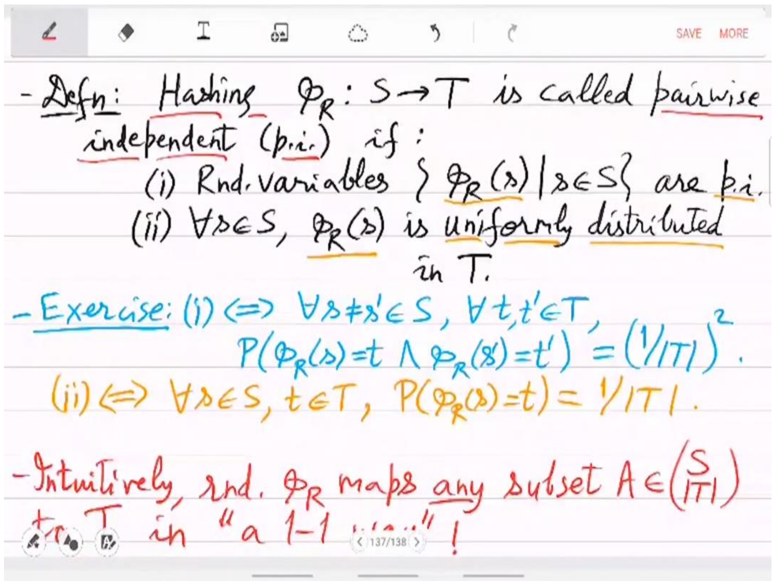
Probability for Computer Science

Prof. Nitin Saxena
Department of Computer Science and Engineering
Indian Institute of Technology - Kanpur

Module - 7
Lecture - 26
Hashing, Introduction to Probabilistic Methods

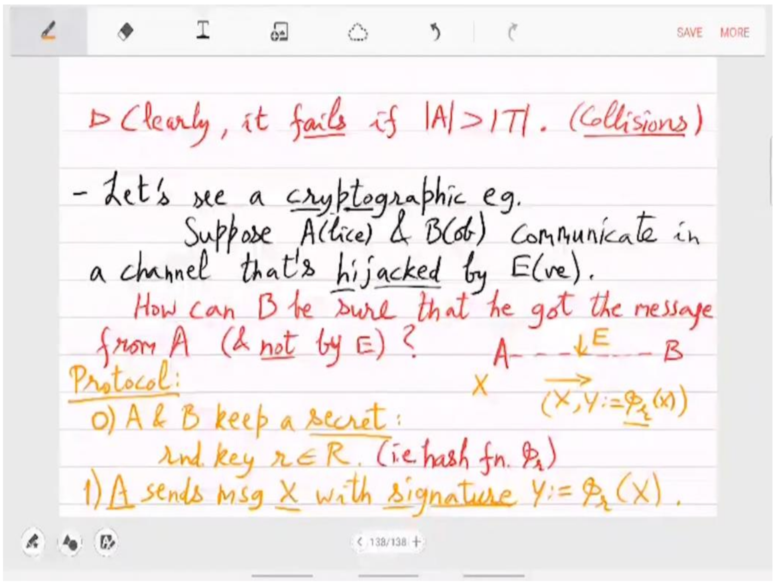
(Refer Slide Time: 00:15)



- Defn: Hashing $\Phi_R : S \to T$ is called pairwise independent (p.i.) if:
  (i) Rnd. variables $\{\Phi_R(s) \mid s \in S\}$ are p.i.
  (ii) $\forall s \in S$, $\Phi_R(s)$ is uniformly distributed in T.
- Exercise: (i) $\Leftrightarrow$ $\forall s \neq s' \in S, \forall t, t' \in T$, $P(\Phi_R(s)=t \wedge \Phi_R(s')=t') = (1/|T|)^2$.
  (ii) $\Leftrightarrow$ $\forall s \in S, t \in T$, $P(\Phi_R(s)=t) = 1/|T|$.
- Intuitively, rnd. Φ_R maps any subset $A \in \binom{S}{|T|}$ to T in "a 1-1 [illegible]"!

So, basically, the goal is that as long as you have picked a big enough T, elements of your interest in S will be bijectively mapped from S to T. Now, this obviously fails if A is a much bigger set.

(Refer Slide Time: 00:32)



▷ Clearly, it fails if $|A| > |T|$. (Collisions)

- Let's see a cryptographic eg.
  Suppose A(lice) & B(ob) communicate in a channel that's hijacked by E(ve).
  How can B be sure that he got the message from A (& not by E)?

$$A \dashrightarrow \overset{E}{\downarrow} \dashrightarrow B \quad \overrightarrow{X} \quad (X, Y := \Phi_r(X))$$

Protocol:
0) A & B keep a secret: rnd. key $r \in R$. (i.e. hash fn. Φ_r)
1) A sends msg X with signature $Y := \Phi_r(X)$.

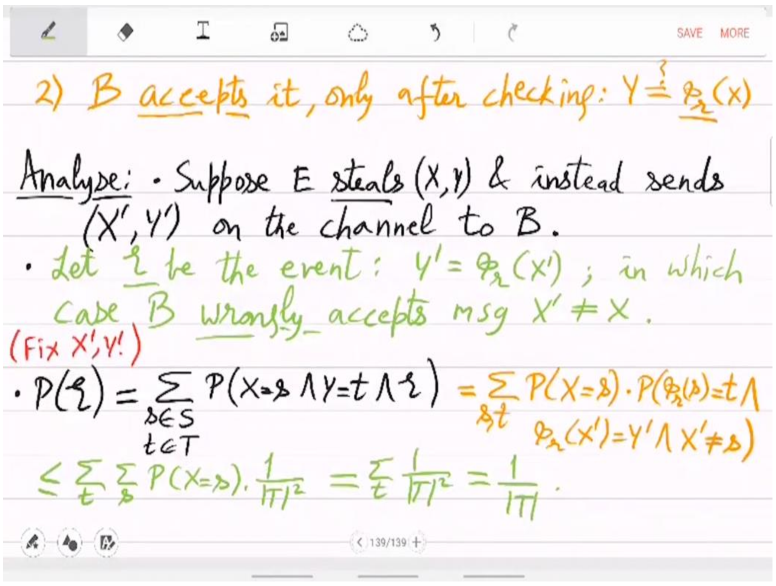
So, clearly, it fails if A is larger than the size of T, because if you are mapping many elements to fewer elements, then there will be collisions. So, you want to avoid collisions as long as we are interested in few elements of S. That is what hashing achieves in a beautiful and very practical way. So, let us see a cryptographic example of why, definition that we have pairwise independent hashing. Why is this so important in practice?

So, suppose Alice and Bob want to communicate by a channel that is hijacked by Eve. So, hijack means not only can E read what is being sent in the channel, but E can also replace that message by her own message; it is completely hijacked. So, given this scenario, how can B be sure that he got the message from A and not by E. So, this is a complicated problem. Again, this seems almost impossible to solve, because B has only one line of communication with A, which is hijacked by E; so, E can do anything; but we can use one loophole, which is A and B may share a secret, which E does not know.

So, with that secret, maybe A and B are in a better position. So, the picture is something like this. There is A; there is B; there is this communication channel which is hijacked by E. So, A wants to send X and some changes have to be made. So, that is X, Y; Y which is hash image of X. So, Alice will not just send the message, but also some kind of a signature, using a hash function.

And hopefully, by analysing the signature, B will know that it is from A and not from E. So, let us see the protocol. In the protocol, A and B keep a secret. So, a random key r, which gives access to hash function; that is a hash function. So, let me use that phi small r. So, that phi r is the important secret that A and B share. And think of it as a signature that B will see from A. Now, why would it work? We have to analyse that. So, let us continue with the protocol first. So, now, using this hash function, A sends message X with signature Y, which is phi r X.

(Refer Slide Time: 06:03)


2) B accepts it, only after checking: $Y \overset{?}{=} \Phi_r(X)$

Analyse: • Suppose E steals (X, Y) & instead sends (X', Y') on the channel to B.

• Let $\mathcal{E}$ be the event: $Y' = \Phi_r(X')$; in which case B wrongly accepts msg $X' \neq X$.

(Fix X', Y')

$$\cdot\ P(\mathcal{E}) = \sum_{\substack{s \in S \\ t \in T}} P(X = s \wedge Y = t \wedge \mathcal{E}) = \sum_{s,t} P(X = s) \cdot P(\Phi_r(s) = t \wedge \Phi_r(X') = Y' \wedge X' \neq s)$$

$$\leq \sum_t \sum_s P(X = s) \cdot \frac{1}{|T|^2} = \sum_t \frac{1}{|T|^2} = \frac{1}{|T|} .$$

And finally, Bob will check; Bob accepts it only after checking whether Y is the same as phi r X. So, note that Bob has X, has Y and also has the hash function. And if this equation Y equal to phi r X is satisfied, then Bob is satisfied that it indeed came from A. Otherwise, he will know that something bad happened, and probably E trapped the original message and sent the corrupted message of own. So, why is this strange protocol supposed to work?

And what is the use of hash, pairwise independent hashing? Let us do that analysis. So, suppose E steals X, Y, and sends some different message X prime Y prime on the channel to B. Now, B would not know a priori that E has stolen the original message. So, B will, in step 2, just check Y prime equal to phi r X prime. So, what happens? So, let E be the event that Y prime is phi r X prime, in which case B wrongly accepts message X prime, which is different from X.

So, let us call it fancy E. This fancy E is the event that B has been fooled by Eve. So, let us do this calculation, probability of this event E happening, when Eve has 1. So, let us go over all the messages and the signature. So, messages s, signature t, the domain in the range of phi. So, s was the correct message that A wanted to send, and it is an image; message is s, signature is t, and bad event happened. So, what is that?

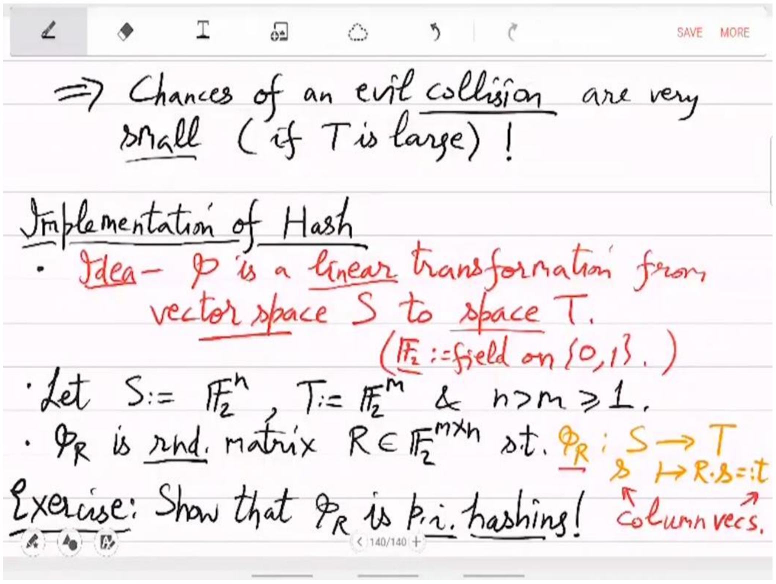
So, we are basically summing over all these bad situations. What is the probability? So, that is, message is s times the probability that phi r s is t and phi r X prime is Y prime, and X prime is different from s. So, this X prime, Y prime is what Bob has already got. So, we are fixing X prime, Y prime, because Bob has already seen X prime, Y prime; but Bob does not know what was X and Y.

So, you are assuming it to be, X to be as string s, and then, the image of that is t, but it so happens that the hash function phi r, it maps X prime to exactly Y prime, even though X prime is different from s. So, this is the sum of errors. Notice that these 3 things in the end, they are independent of X being s. So, we have written as a product, because phi r is that the, it does not depend on what the message was, and X prime also does not depend on that; I mean, X prime is different from s; so, this is independent; that is what we are assuming.

So, this sum of errors is less than equal to sum over t, sum over s, probability that X is s times. So, what is the chance that s goes maps to t and X prime maps to Y prime? So, this, by the property of pairwise independent hashing, is exactly 1 over T square. This is where we are using pairwise independence. And we are also using the fact that s and X prime are different. If Eve used s equal to X prime, then, there is no problem.

The error only happens when X prime is different. So, that is the case which pairwise independence property also captures. So, X has to be one of the s's; so, this sum is 1. So, you get sum over 1 over T square, which is 1 over T.

(Refer Slide Time: 13:13)



$\Rightarrow$ Chances of an evil collision are very small (if T is large)!

Implementation of Hash

- Idea – φ is a linear transformation from vector space S to space T.

  ($\mathbb{F}_2 ::=$ field on $\{0,1\}$.)

- Let $S := \mathbb{F}_2^n$, $T := \mathbb{F}_2^m$ & $n > m \geq 1$.
- φ_R is rnd. matrix $R \in \mathbb{F}_2^{m \times n}$ s.t. $\varphi_R : S \to T$

  $s \mapsto R \cdot s =: t$

  column vecs.

Exercise: Show that φ_R is p.i. hashing!

So, which means that chances of an evil collision are very small if T is large. So, this is what the hash function did. Since E will not know what the hash function is, she will not be able to find X prime, Y prime so easily. So, as long as the image of the hash function is large, the range is large, probability of collision being discovered by Eve is very little. Maybe I will also quickly talk about how a hash function is implemented.

So, at this point, you may not even believe that it exists. We have only defined it, and then we saw an application; but should it exist? So, let us see a quick idea to implement. So, let us think of S and T as vectors. In fact, they are vector spaces. So, S has a big ambient space; T has a small ambient space. And then, the map is a linear transformation. So, phi is a linear transformation from vector space S to space T.

So, this is what is best for implementations, linear transformations and vector spaces. So, phi in other words, we will implement as a matrix. And then, messages and the signatures will be vectors. So, how? So, more concretely, you can take S to be 0 1 to the n. So, F 2 is the field of 2 elements. So, F 2 is this binary field. So, when you add, you add basically mod 2. And when you multiply, you multiply mod 2.

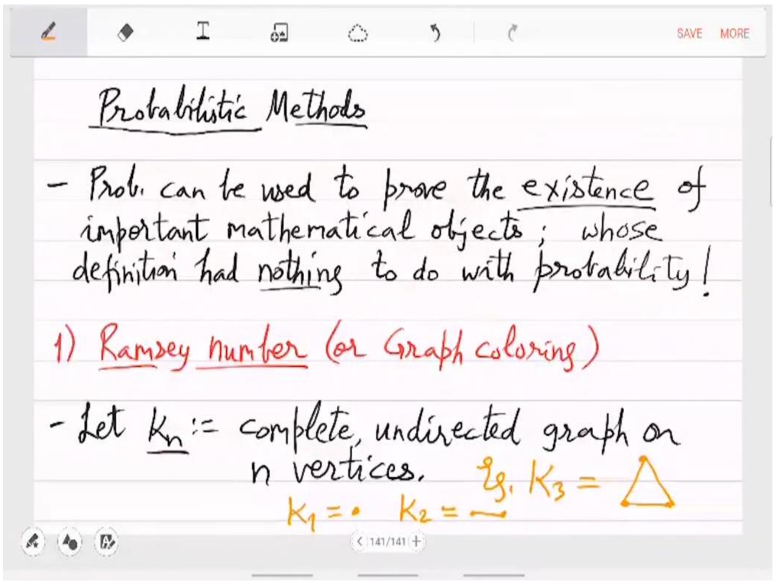
So, 1 dot 1 is 1, and 0 dot 1 and everything else is, all the other products are 0. And 1 + 1 is also 0. So, with that understanding, now, bits have turned into algebraic objects. So, S is F 2 to the n; T is F 2 to the m; and n is bigger than m. So, large space to small space. Now, phi R is a random matrix R. So, it is m cross n, such that this action phi R from S to T is simply given by, column vector S will transform to column vector by left multiplication.

s is a column vector with n coordinates, and when you left multiply by R, it becomes a column vector with m coordinates. So, a long vector, like, bigger ambient space vector into a smaller ambient space vector; this is what is happening. And obviously, you just pick a random matrix by flipping mn coins or a coin mn times. You will get this, all these entries of R field.

So, it is very easy to randomly sample and it is equally easy to define the action and do this computation. So, I will leave it as an exercise. Show that phi R is pairwise independent hashing, which means that, as you randomly pick these matrices, m cross n matrices, these 2 conditions that you had in the definition like pairwise independence and uniform distribution of the image, both of them will be satisfied.

They will basically be, I mean, think of it in terms of a single row, the first row of R. So, first row has at least 2 random bits, 2 elements. So, when you multiply with s, you can see that; with s and with s prime, you will basically get 2 linear equations; and those values will be independent. So, this can be done. This is a very useful construction, and it is also done in future courses. So, I skip this and I move to a new topic, the last one in this course, which is probabilistic methods.

(Refer Slide Time: 20:10)



Probabilistic Methods

- Prob. can be used to prove the existence of important mathematical objects; whose definition had nothing to do with probability!

1) Ramsey number (or Graph coloring)

- Let $K_n :=$ complete, undirected graph on n vertices.

Eg. $K_3 = \triangle$

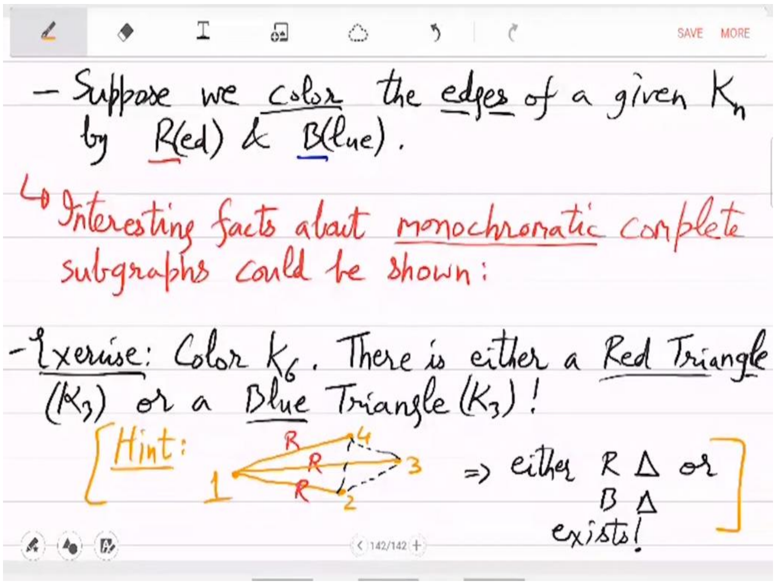
$K_1 = \bullet$ $K_2 = —$

So, you may ask, what is new in this title? We have been doing probabilistic methods; we have been doing probability all the while; so, difference is that probabilistic methods, we will use this term to mean that we are using probability to show that some mathematical object exists. It is for the existence. It is not really to compute probability, but to even show existence, just show existence.

Probability can be used to prove the existence of important mathematical objects, whose definition, nothing to do with probability. We will see many examples of mathematical objects whose definition is completely combinatorial, there is no probability; and yet, by using probabilistic methods, by using probability, we show that they even exist. The existence was not clear from the definition. So, that is the surprise in all these applications.

So, instead of talking about this in more general terms, let us see the examples. First is Ramsey number or graph colouring. So, let K n be the complete undirected graph on n vertices. So, for example, K 3 is simply a triangle, and K 4 will be a square with both the diagonals, and so on. K 2 will be a line segment, and K 1 is just a point, just a vertex. So, what we want? We want to look for such subgraphs, graphs or subgraphs of the type K n, in coloured graphs.

(Refer Slide Time: 24:07)



– Suppose we color the edges of a given K_n by R(ed) & B(lue).

↳ Interesting facts about monochromatic complete subgraphs could be shown:

– Exercise: Color K_6. There is either a Red Triangle (K_3) or a Blue Triangle (K_3)!

[Hint: $\Rightarrow$ either R Δ or B Δ exists!]

Suppose we colour the edges of a given graph G by red and blue; so, given a graph G, we have arbitrarily coloured the edges, only the edges red or blue. So, we are basically, in other words, partition the edges into 2 types, then what happens? Then, can we identify these complete subgraphs and monochromatic? In fact, let me say that it is K n that we are covering. Then, interesting facts about monochromatic complete subgraphs could be shown.

This is what we are after. So, K n, we have coloured the edges. So, now, we will say that, okay, there will either be a red or a blue monochromatic subgraph that is again a complete graph. So, for example, just to warm you up; colour K 6, complete graph on hexagon. There is always either a red K 3, which means triangle; there is either a red triangle or a blue triangle; show this.

So, no matter how you colour K 6, the edges, you will not be able to avoid red triangles and blue triangles together. So, let me give you a hint, how this will be done. And then, maybe you complete it. So, focus on vertex 1; there are 6 vertices. Now, vertex 1 has 5 neighbours which have been coloured, these edges have been covered red or blue. So, 3 of them have the same colour, let us say red.

So, 2, 3, 4; they are all coloured red. So, either there you will find 3 neighbours, 3 outgoing edges which are red or 3 that are blue. That happens by simply just averaging argument or pigeonhole principle. And now, what can you say about these 3 edges, 2, 3; 3, 4; 4, 2? So, either one of these edges is red, in which case you get a red triangle or all of them are blue. So, this implies either red triangle or blue triangle exists. So, that is your hint. So, next time, we will generalise this.