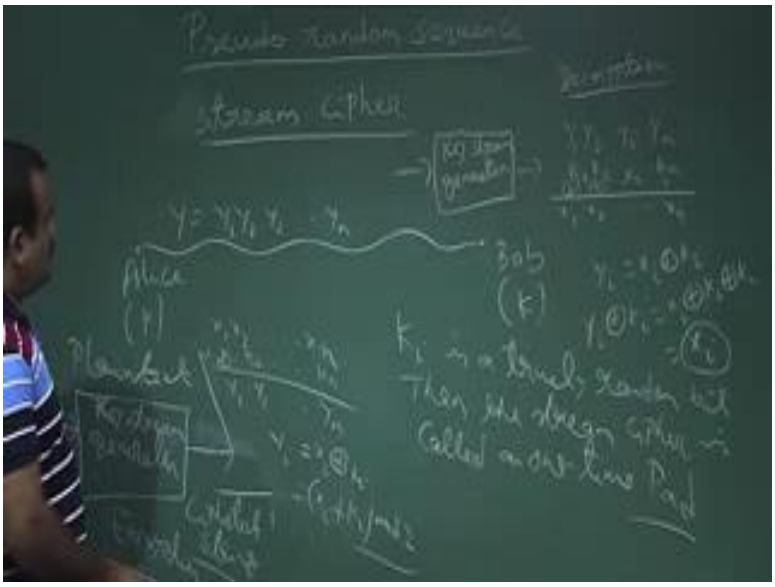
Internetwork Security
Prof. Sourav Mukhopadhyay
Department of Mathematics
Indian Institute of Technology, Kharagpur

Lecture – 10
Pseudorandom Sequence

We will talk about pseudorandom sequence, this is basically this is the neat field of in the stream cipher which is a symmetric key primitives.

(Refer Slide Time: 00:34)



Let us just recall the stream cipher. So, basically in stream cipher, what we have this is a symmetric key encryption. So, Alice and Bob they agreed with a common key k and in stream cipher, suppose the Alice, the plaintext is in binary form, plaintext in binary bit, binary bits 01 bit say 010110 like this. So, this is a, so now we have this key. So, what Alice will do? Alice will run a; what is called random number generator. So, this is a key stream generator; key stream generator which takes the input as this secret key and we should able to generate the key bits, 01 bits again. So, 10110 like this. So, this is the key bit. So, key stream, key stream in 01 bits.

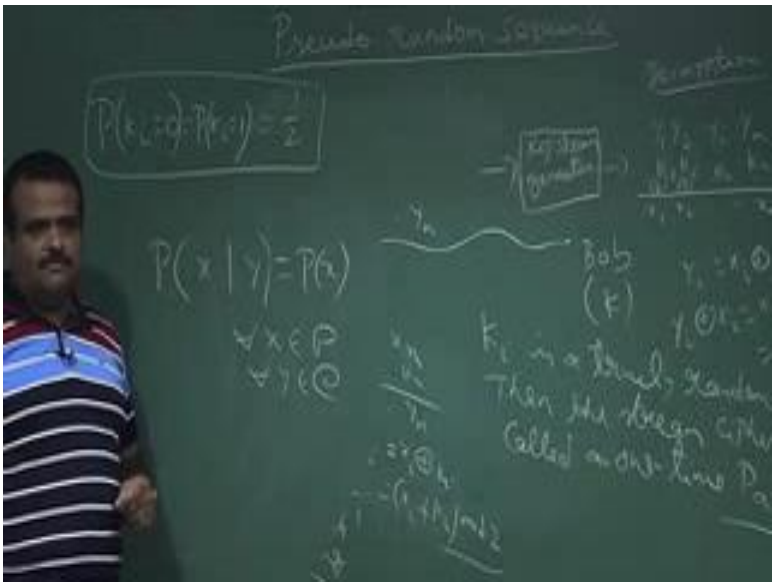
And then to get the cipher text what we do? We just do the bitwise extract. So, this is 1, this is 1, this is 1, this is 0, no carry, just the bitwise XOR, like this. So, this is the cipher text stream. So, this is the cipher text stream. So, basically this cipher text stream, so Alice will send to Bob.

So, if this is if the plaintext stream is say x 1, x 2, x n. So, if the plaintext stream is say x 1, x 2, x n, if there are n bits 01 bit and so, 1 has to get the key stream generator which should give us the key stream k 1 k 2 k n and we do the bitwise excluding and we got this y 1, y 2, y n. So, y i is equal to x i, x 1 with k i. So, just the bitwise extract, so this is basically x i plus k i mod 2 and this y 1, y 2, this is basically cipher text y n is basically cipher text. So, this is the encryption, this is what Alice is doing is encryption these, we are talking about stream cipher and then Bob; how Bob will decrypt it?

So, Bob is receiving y 1, y 2, the cipher text stream and so, Bob is having same key k, so Bob also have this key stream generator, this function is public, only secret is the secret key. Stream generator, the same key stream generator, so that it should get the same input as the initial value, it should give us the same key stream. So that k 1, k 2, k n so that if you just do the bitwise XOR again it should give a x 1, x 2, x n because this y i is basically x ix of ki, now again with this y i we are doing x o r with this is y i and k i. So y, this is Bob is doing y i x of k i is basically x i x of k i x or k i, so it should give us x i. So, this is the ith bit of the plaintext. So, this is the decryption what Bob is doing.

Now, we have already seen that depending on the key stream, this x i; our security will be I mean this key cipher will be more secured if this is a truly random sequence, if k i is a truly random bit; truly random bit random bit then this is a basically this satisfy the Shannon notion of perfect secrecy and which is basically one time pad then this stream cipher is a onetime pad then the stream cipher is an one time pad which process which process the Shannon notion of perfect secrecy.

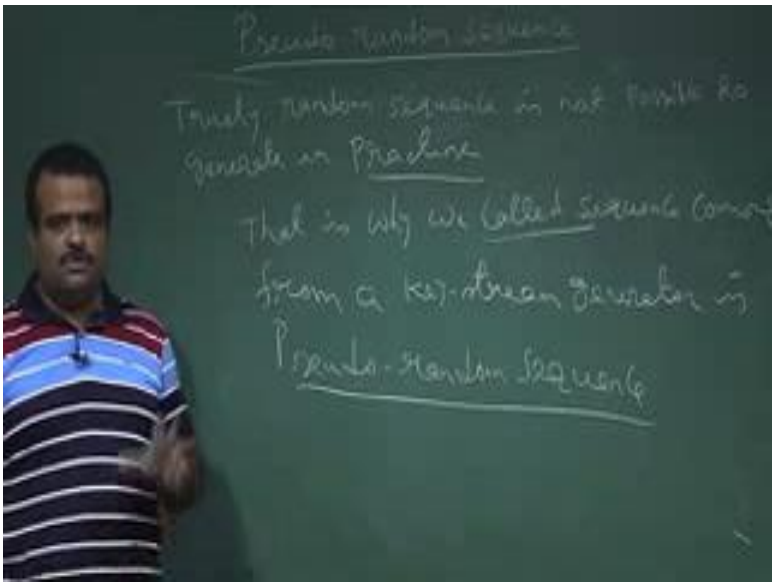
(Refer Slide Time: 06:15)



The truly random bit means this probability of k i is 01 bit. So, it is either 0 or 1 bit. So, probability are equal probability of 0 and probability of 1 should be equal then it is called truly random sequence and we have seen in the earlier class and then if this satisfy then its process Shannon notion of perfect secrecy. That means, given the cipher text the probability of, so probability of given the cipher text probability of the plaintext is same as probability of plaintext this is the Shannon notion and this is true for all x, all the plaintext this is called Shannon notion perfect secrecy all the plaintext and all the cipher text.

Cipher text space, so, if this is satisfy then we are not getting any information by seeing the plaintext because plaintext is public, it is just send over the public channel. So, Oscar is also having the excess of the plaintext. So, now, if this probability is same as this probability of the plaintext; that means we are not getting any advantage of knowing the cipher text and by assuming this is truly random bit we have seen in the last class that we can achieve the Shannon notion of perfect secrecy. But the problem is to get a truly random base is not practical because we have to follow some algorithm deterministic algorithm to generate this random numbers so that getting a truly random numbers is not practical it is not possible in practice so that is why we call this is a pseudorandom sequence instead of truly random sequence we call this sequence x 1, x 2, x n because truly random sequence is not possible.

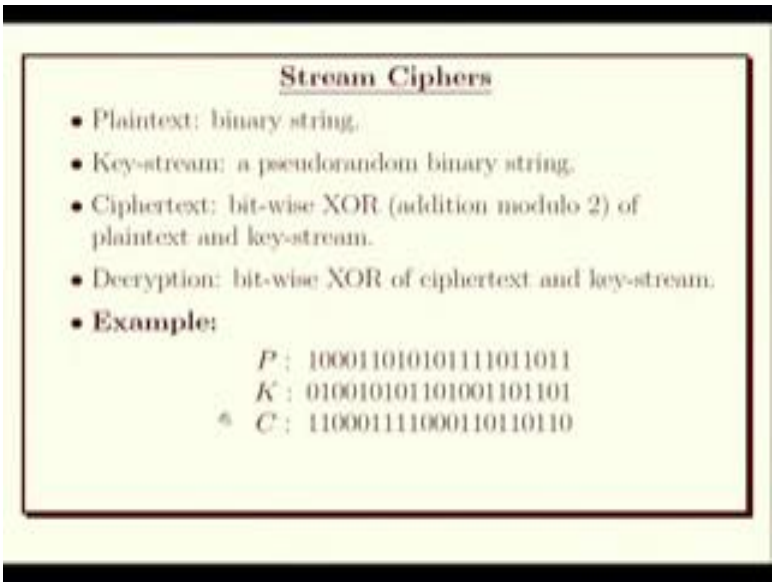
(Refer Slide Time: 08:25)



We want to be a truly random sequence, but truly random sequence is not possible to generate sequence is not possible to generate in practice. So, it is basically impractical to have a two generate truly random sequence. So, that is the reason we call this sequence which we are generating from the key stream generator is called pseudorandom sequence that is why, we call the sequence coming from a key stream generator; generator is called that is why we call the sequence coming from a key stream generator is pseudorandom sequence instead of truly random sequence, pseudorandom sequence.

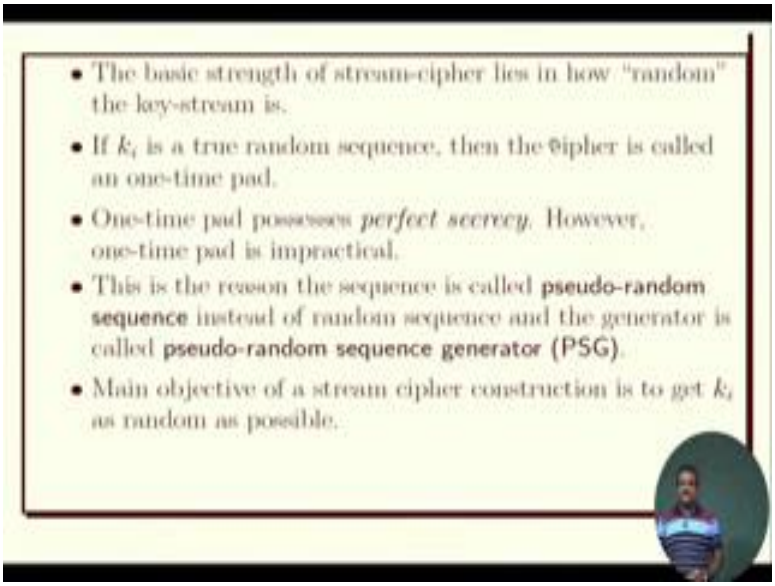
And in this sequence we want as much randomness as possible. So, we want this sequence should be close to a truly random sequence; that means, we want to have some test on this sequence which we are getting from a key stream generator and it should satisfy those test. So, those are all statistical test. So, those are basically the necessary property a truly random sequence should have. So, basically we are from the key stream generator we are getting a sequence that we are telling pseudorandom sequence because it is not a truly random it is a pseudorandom. So, now, is a stream cipher will be a good stream cipher if the sequence is close to a truly random sequence. So, the security of the stream cipher will depend on how good this randomness is there in the sequence.

(Refer Slide Time: 11:07)



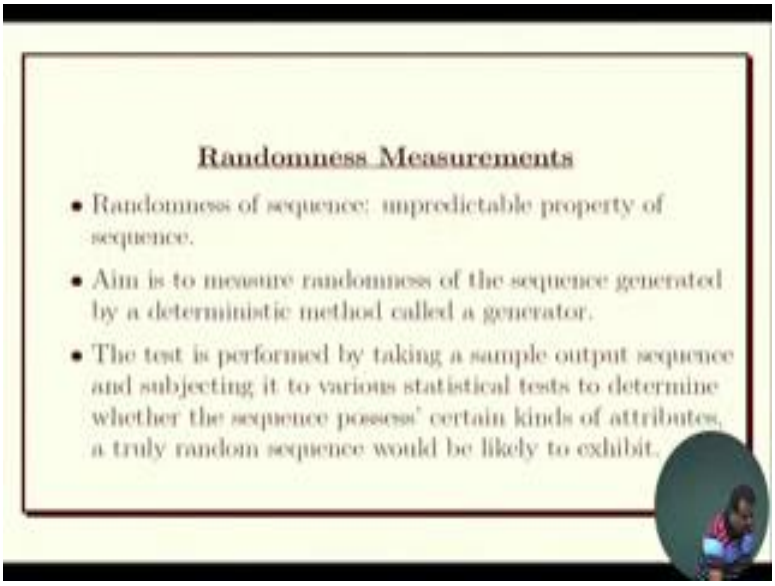
The security of the stream cipher will depend on how good this; can you please go to the slide yeah. So, this is the stream cipher in the stream cipher we typically have a plaintext as a binary string and we need to generate the key stream for which we have a key stream generator will we will take an example of a key stream generator we have seen LFSR linear feedback shift register is the key stream generator. And then to get the cipher text bit we just XOR the plaintext bit with the key stream and we get the cipher text and decryption is the reverse way, we just again XOR with the key stream and we got the cipher text plaintext back.

(Refer Slide Time: 11:42)



Basic strength of a stream cipher lies in how random the key stream is because we know that key stream cannot be truly random sequence. So, now, question is how random this is. So, if k i is truly random sequence then the cipher is called 1 time pad. So, that posses the Shannon notion of perfect secrecy; however, one time pad is not possible to have in practice this is impractical. So, this is the result. So, we call this sequence as truly random sequence instead of, we call this sequence as pseudorandom sequence instead of truly random sequence and the key, stream generator which is giving this sequence is called pseudorandom sequence generator. So, the main objective of the stream cipher is to construct this key stream as random as possible.

(Refer Slide Time: 12:34)



So, we have to do some randomness measurement of this sequence, suppose we have a key stream generator and which is giving as a key stream.

(Refer Slide Time: 12:46)

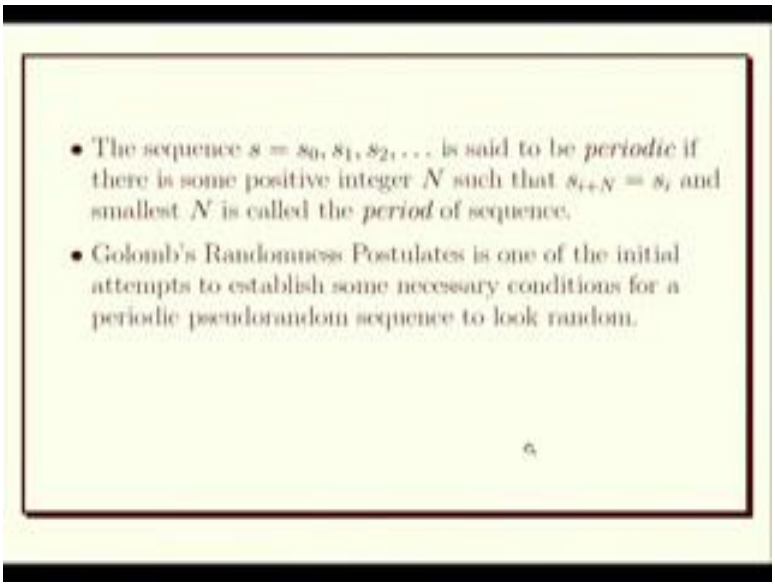
Say this is a key stream generator or we can now say pseudorandom; pseudorandom sequence generator basically this is a key stream generator which is taking the secret key k and it is giving us the key stream. So, k 1, k 2, k n, how many key stream we need, but it has a period, we will come to that what do you mean by period of a key stream. So, so this is the. So, now, we want to see how good this sequence is, how good this sequence is. So, we take a period of this sequence and we do some randomness test on this sequence. So, how random this sequence is so will do the randomness measurement on this sequence. So, randomness means unpredictable property of the sequence.

Our aim is to measure the randomness of the sequence generated by the deterministic method which is called the pseudorandom bit generator and so, we need to perform some test by taking sample output of the sequence, basically we take a period of the sequence and then we run those test. So, those test are basically coming from those are the necessary property, a truly random sequence should have. So, those tests are basically necessary test, but if we say that we are suppose we define 10 tests and our sequence is satisfying all the 10 tests then also we cannot say that this is a truly random sequence because there may be some other property, 1000s of property truly random sequence should have. So, these conditions are necessary conditions.

When you say sequence is a good random sequence, if it pass all the test, we know, but there will be other test, there may be other property if truly random sequence should

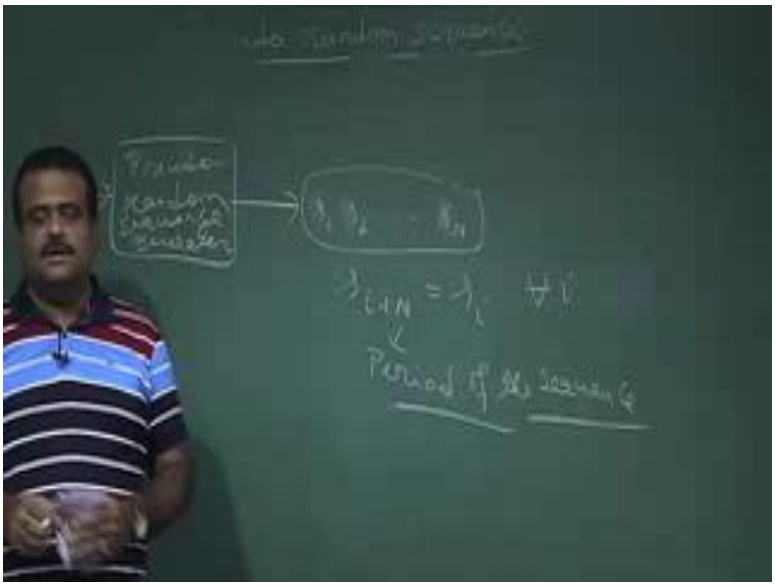
have. So, this condition are typically necessary condition not the sufficient condition; that means, if we pass all the test, I mean all the test means whichever we have there may be other high wanted test which we do not know yet. So, if it satisfies all the test then also we say this is little bit of good sequence because it is satisfying all the test known to us, but still we cannot say this is a truly random sequence because there may be 1000s of other test which we do not know.

(Refer Slide Time: 15:22)



This is for this test, what we do? We take a sequence and we say this sequence is periodic if after n it is repeating the bit. So, this n is called period of the sequence.

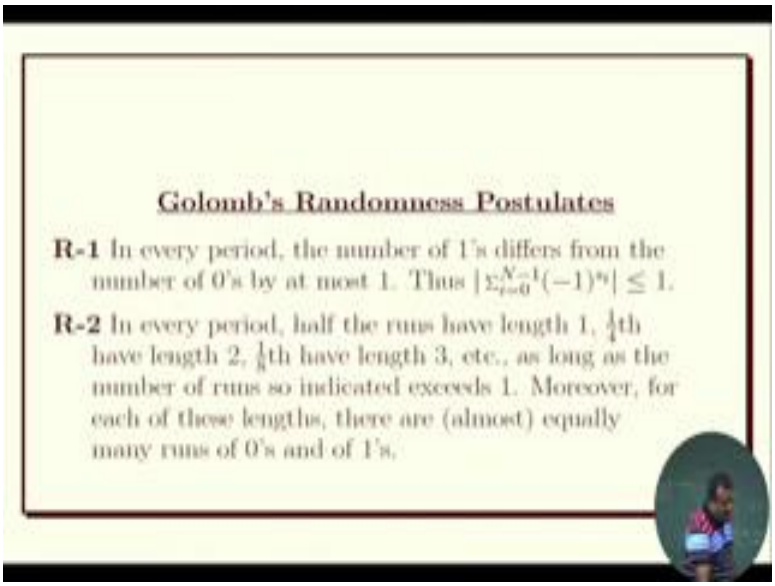
(Refer Slide Time: 15:45)



We take a sequence like so after n this is k i. So, if you take this in the slide, we take s i. So, anyway does not matter. So, if s i plus n equal to s i for all i so; that means, after n iteration after n we have a repetition so; that means, this is a period. So, we take a period. So, n is called this is the minimum integer n, n is called minimum value of such n is called period of the sequence.

Now we take a period of a sequence and we apply the Golomb's postulates which are the necessary condition to a sequence to be a random sequence truly random sequence those conditions are necessary condition. So, let us talk about Golomb's randomness property say it has basically three rules, a postulates.

(Refer Slide Time: 16:42)



First one is telling. So, every period in every period; we have exactly same number of wants of 0, I mean if that effect they differ by only 1. So, if you take a period like.

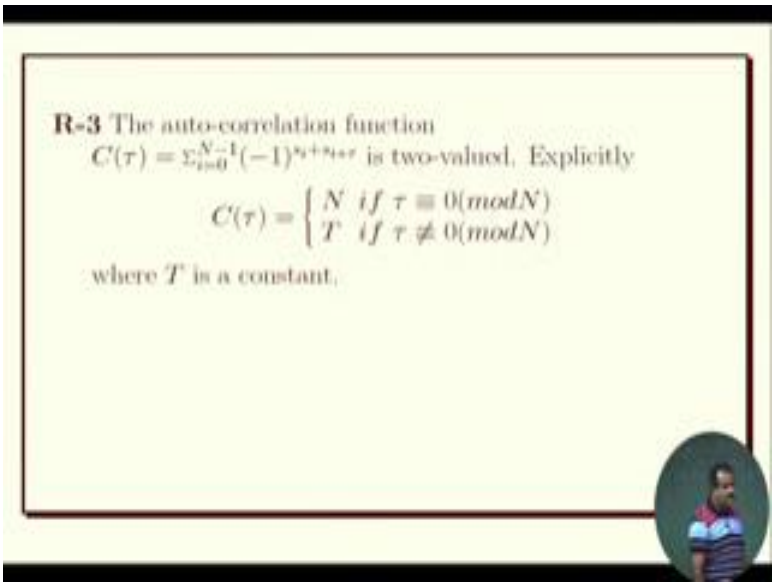
(Refer Slide Time: 17:12)

So, this is Golomb's postulates this tests are by Golomb - Golomb's randomness test; what it is telling? It is telling, it has 3 rules, 3 postulates, first one which is telling the we take a sequence s 1, s 2, s n, we take a sequence and the first it is telling that we count the number of 0s number of 1s and they must be almost equal if they differed then they will be differed by 1 so; that means, cardinality of number of 0s minus cardinality of number

of 1s must be less than equal to 1. So, that is the first postulates. So, in every period the number of 1 differed from the number of 0 by at most 1. So, this way we can see that and if we consider the run in every period half of the run has length 1, one-fourth of the run should have length 2, one-eight, I have run of length 3 and so on. This is the second condition; this is about run of a sequence run means the consecutive bits.

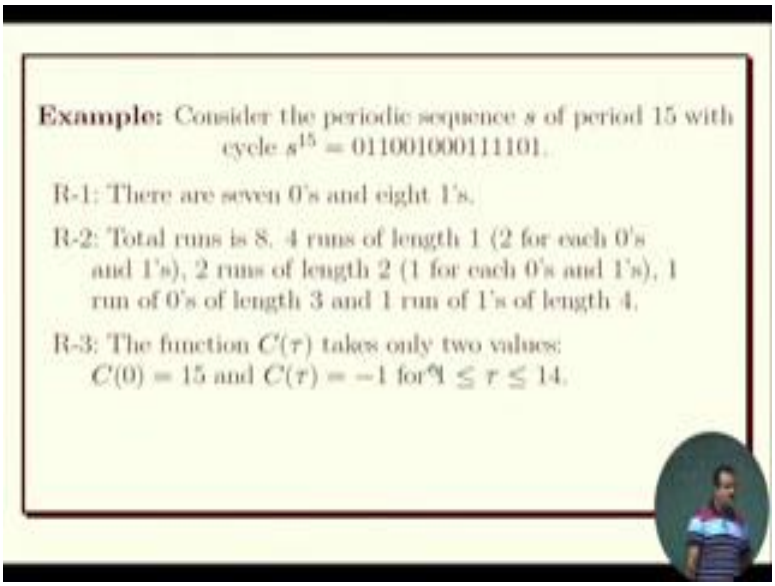
We will take an example. So, we count that how many runs are there. So, if there are 8 runs then half of the run should be of length 1 so; that means, there are 8 runs, so 4 runs should be of length 1 and one-fourth run should have length 2 like this. This is second condition or second rule about the run, we will take an example

(Refer Slide Time: 19:17)

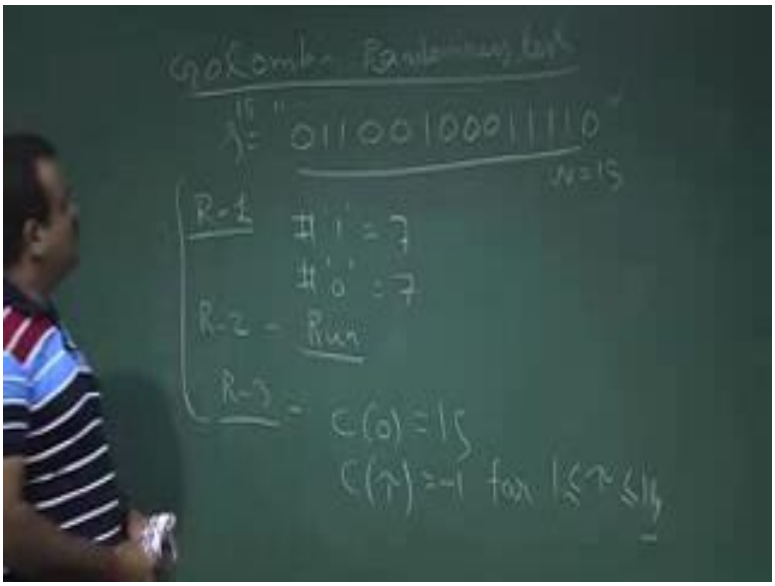


And third rule is auto correlation function. So, we define this function C tau which is basically we take a x i and we take x i plus tau and we take this relationship and we define this for tau is equal to 0 to N and if this is a constant function it is N if f T is constant, it is N, if tau is a multiple of N, if it is 0 or if it is N to N otherwise it should be a constant. So, this is auto correlation function, if this is a constant function then we say the third rule of the Golomb's is satisfying. So, now, let us take an example.

(Refer Slide Time: 20:02)



(Refer Slide Time: 20:11)



Suppose we consider a sequence like this, suppose our sequence is say 11 sorry, 01100100011110. So, this is a period of a sequence coming from a 0 random bit generator, it could be LFSR we will come to know, it is basically coming from a LFSR. So, now so we will apply, so this is our sequence s, a length, this is of period is 16. So, N is 16, N is 15 over here. So, after that it will repeat over here. So, we take a period of that sequence. Now rule number 1, so we count the number of 1s and number of 0, what is the number of 1s? 1 2 3 4 5 6 7, so number of 1s is 7 and what is the number of 0s? 1 2 3 4 5 6 7, so it is basically same. So, they have the same number of if they differ they will

be differed by at most 1. So, if the number so here it is exactly same. So, it is satisfying the first rule.

And the second rule is telling the about the run. So, the total run is 8, run means we just count the subsequent digit. So, this is 0, this is run of length 1. So, this is 11, this is run of length 2, again we have a run of length 2, then we have a run of length 3, we have a run of length 4 and we have a run of length 1. So, if you just count the total number of run, there are 8 runs. So, among this, this rule is telling half of the run should be of length 1 so; that means, half means 4 runs should be of length 1. So, 4 run 1 2 3 4, this is of length 1 and one-fourth of the run should be of length 2, one-fourth of 8 is basically 2. So, 2 of the run should be length 2. So, this 1 length 2 run, this is 1 length 2 run and one-eighth of the run should be of length 3, one-eighth means only 1. So, this is the 1 like that. So, this is satisfying this run is satisfied. So, this is the rules regarding whether run is satisfied.

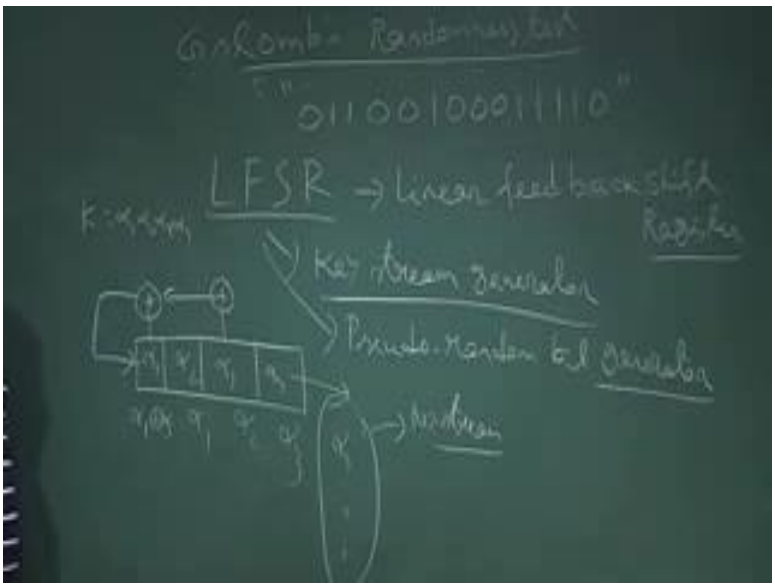
Now, we consider rule number 3 which is basically auto correlation function. So, if you define this correlation function tau 0 which is this formula we will be using. So, C tau is equal to this, we take s i then s i plus tau. So, if tau is 0 then this is basically s i twice if tau is 1 then s i, s i plus 1 like this. So, if you calculate this it will give us a constant function where C 0 is 15 and C tau will be minus 1 for 1 less than equal to tau less than equal to sorry 14. So, this is the auto correlation. So, this is a constant function. So, this Golomb's postulates satisfy for this, this sequence.

Now we will see where from this sequence is coming. So, we know the, so there may be other test this is Golomb's test. So, there may be other test which is basically the necessary condition a truly random sequence should have, like we may have test like we just count here - we have counted the number of 0s number of 1.

(Refer Slide Time: 24:25)

We can count the number of this bit 00011011, we count their number and they must be similar. So, this is another property of a truly random sequence. So, there could be many properties of a truly random sequence. So, those are basically and there are many statistical tests on this. So, those are basically necessary test. So, a truly random sequence should pass this test. So, if you have a pseudorandom sequence it should pass this test then it will be having more randomness now talk about how we can get this sequence. So, we know this LFSR we have used as a pseudorandom bit generator.

(Refer Slide Time: 25:31)



We will see how the LFSR bits are, LFSR generates this, how good sequence LFSR is generating. So, will talk about LFSR which is basically Linear Feedback Shift Register and so, it is basically a key stream generator or pseudorandom bit generator. It is basically a key stream generator, it will generate the key stream that is why it is called pseudorandom bit generator and so, it is basically if we just have a 4 bit LFSR, we can just have 4; 4 register and it has some connection, the connection could be anywhere in the say connection could be here, connection could be here, like this. So, this is one example like this. This is a 4 bit LFSR.

So, this is basically whatever the bits over here and here will just take this out will XOR and we will put it here as a linearly shifting we are shifting this bits. So, this is if the k is shared key is alpha 1, alpha 2, alpha 3, alpha 4 which is the secret key shared between Alice and Bob, this is 4 bit. So, you just put these values and then we run this LFSR. So, then in the first shot this alpha 4 will be coming out this is a bit 0 or 1 either and this will go here, this is linear feedback shift register, this will come here this will come here and this bit will be basically XOR between this and this. So, this is basically alpha 1 XOR alpha 3. So, this way, we continue and will get a key steam.

So, this is key stream and this key stream is basically pseudorandom bit; pseudorandom sequence and will see the connection between LFSR and the polynomial and if we chose a good polynomial like primitive polynomial then it will give us a good sequence. So, it is a good pseudorandom bit generator depending on the choice of the polynomial. So, those will discuss in the next class.

Thank you.