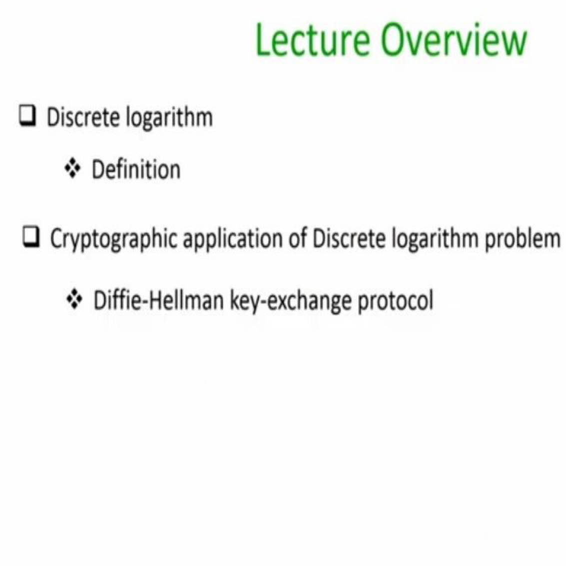
Discrete Mathematics
Prof. Ashish Choudhury
International Institute of Information Technology, Bangalore

Lecture - 65
More Applications of Groups

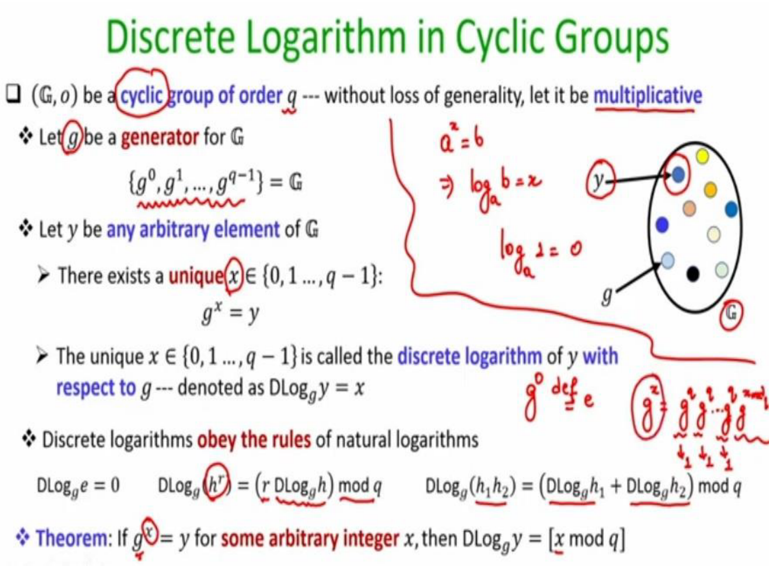
Hello, everyone, welcome to this lecture. So, till now we have discussed a lot of theory regard in the context of number theory and abstract algebra. In the next couple of lectures we will see how to tie whatever we have learned till now in number theory and abstract algebra and see some concrete applications in the context of cryptography.

(Refer Slide Time: 00:42)



So, the plan for this lecture is as follows. In this lecture, we will introduce this concept of discrete logarithm and the discrete logarithm problem. And we will see some cryptographic applications of the discrete logarithm problem namely the seminal key exchange protocol due to Diffie and Hellman.

(Refer Slide Time: 01:01)



So, let us start with the discrete logarithm definition in the context of cyclic groups. So, let G be a cyclic group and that abstract operation is o and suppose the order of the cyclic group is q that means, you have q number of elements and for simplicity and without loss of generality, I will follow that multiplicative interpretation, while giving the definition of discrete logarithm, but the definition can be easily generalized even when the underlying operation is interpreted in the additive sense.

So, since my group G is a cyclic group it must be having a generator, so, let the g be generator and since the order of the group is q, has q number of elements that means, by raising or by computing q different powers of the generator, I can obtain all the elements of my group. Now, consider an arbitrary element y from the group, since the element y is a member of the group, it can be generated by some power of your generator.

That unique power in the range 0 to q - 1 which when raised to the generator gives you the element y, will be called as the discrete logarithm of the y to the base g. That is the definition of my discrete logarithm. So, in some sense, it is equivalent to our definition of natural logarithms. So, we know that if a to the power x = b then we say that log of b to the base a is x, we are trying to come up with an equivalent definition in the context of a cyclic group.

So, g is a special element the generator because, if I keep the generator to the base and compute different powers of the generator, I can obtain all the elements of my group that means, I can say that you give me any element of the group there must be some power of the generator such that that power x of the generator gives me the element y, that unique power x in the range 0 to q - 1 is called as the discrete logarithm.

And interestingly, like the natural logarithms, your discrete logarithm also obeys the rules that are there in the context of natural logarithms. For instance, we know that log of 1 to the base of any a is 0 because a to the power 0 is defined to be 1 in for natural logarithms. In the context of discrete logarithms, we say that the discrete log of the identity element of the group to the base of g will be 0.

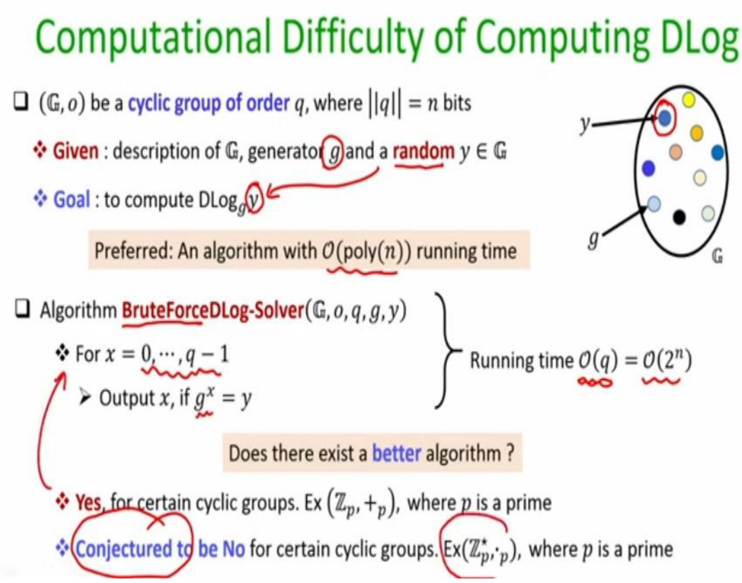
Because, remember, as per the rules of group exponentiation, we have defined g^0 to be the identity element. In the same way, if I take an element h from the group and compute the element h to the power r which will be also a group element and now if I try to find out the discrete logarithm of the group element h to the power r to the base g then it will be same as r multiplied with the discrete logarithm of h to the base g.

And if this value is in the range 0 to q - 1 well and good, otherwise, I take a mod and bring down the value to the range 0 to q - 1. In the same way, if I have 2 group elements h_1 and h_2 then the discrete log of the product of h_1 and h_2 will be same as the summation of the discrete logs of h_1 and h_2 individually modulo q. And the general theorem statement that we can have is the following.

If you are given that $g^x = y$ where g is the generator then, either x will be the discrete log of y if x is within the range 0 to q - 1, else if x is greater than q then, if I take x modulo q then the resultant value will be in the range 0 to q - 1 and that will be the discrete logarithm of y to the base g. This is because if x is indeed greater than q then I can rewrite g^x as several blocks of g^q, g^q, g^q and the last block consisting of $g^{x \ mod \ q}$.

And each of these blocks of g^q will give me the identity element because g is the generator and its order will be q, hence g^q will be identity element and the last block will be $g^{x \ mod \ q}$.so, I get $g^{x \ mod \ q}$. So, if $g^x = y$, so, is $g^{x \ mod \ q}$ and $x \ mod \ q$ will be a value in the range 0 to q - 1 and hence it will be the discrete logarithm.

(Refer Slide Time: 06:17)



So, now an interesting problem is that how easy or how difficult it is to compute the discrete logarithm. So, imagine we are given an abstract cyclic group of order q and this notation means that, the number of bits that I need to represent my q is n bits. So, this notation is nothing but the number of bits needed to represent q. That means magnitude wise q is as large as 2^n. Now, let us see how difficult or how easy it is to compute a discrete logarithm.

So, you are given the description of the cyclic group. By the description I mean, you know, the characteristic of the elements of the group, your group might be exponentially large. It might have exponentially large number of elements and you may not have sufficient space and resources to store down all the elements of your group. But you may know the characteristic or the properties of the elements of your group.

And you are given a generator. So, you know that, by computing different powers of that generator, you can generate any element of your group. And what is given to you, a random element y from the group. That is important, a random element. It is not a predetermined or specific element of the group, it is randomly chosen. And a discrete log problem is to compute the discrete log of this randomly chosen y, given that, you only have the y and the generator, you do not have anything else.

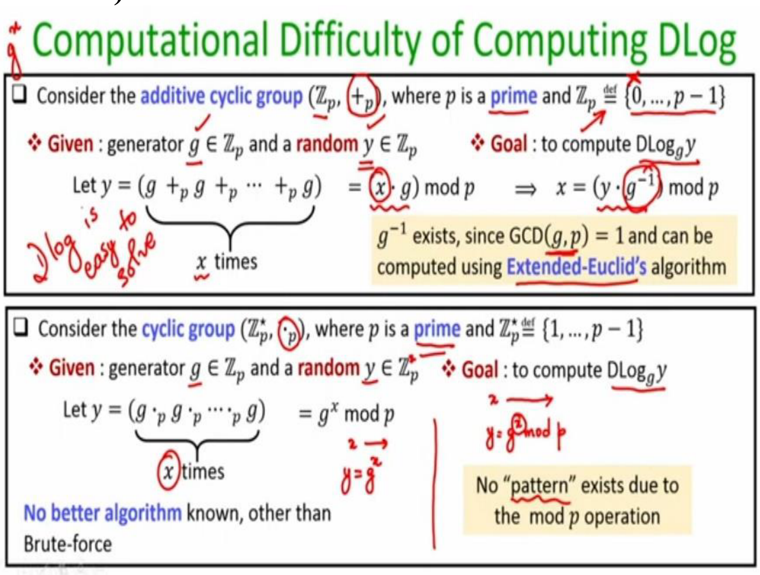
So that means your goal is to come up with a unique power x in the range 0 to q - 1 such that, g^x would have given you y. And what we want to do is, we want to come up with an algorithm whose running time should be polynomial in the number of bits that I need to represent my q, namely, n, I do not need an exponentially large algorithm. So, here is a naive algorithm which will always be successful to give you the discrete log of the randomly chosen y.

I call this algorithm as brute force discrete log solver because it basically does what a naive algorithm will do, you basically try all powers of x in the range 0 to q - 1. And check whether computing g power x gives you the element y or not, if it is then, you output that x and stop the algorithm. And definitely you will hit upon the exact value of x which is the discrete logarithm of y somewhere when you are iterating over all values of x.

So, you will always get the answer. But let us focus on the running time of this algorithm. In the worst case, you may end up performing iteration over all candidate values of x. So, I can say in the worst case, the running time is order of q. But q is not a polynomial quantity in n, it is actually an exponentially large value in the number of bits that I need to represent my value q. So, this algorithm is not a polynomial time algorithm but rather it is an exponential time algorithm.

So now, the next question is does there exist a better algorithm? And the answer is both yes, as well as no. Yes because as we will see soon, there are indeed certain cyclic groups where I can efficiently find out the discrete logarithm of any randomly chosen y, without doing the brute force. But at the same time, we do have some candidate cyclic groups where it is for which it is conjectured that we do not have any better algorithm other than brute forcing over all candidate values of x. So, this is a instance of one such cyclic group.

(Refer Slide Time: 10:39)



So now, let us see the case where the discrete log can be easily computed. So, I consider the cyclic group $\mathbb{Z}_p$ where $\mathbb{Z}_p$ is the set of all integers modulo p, namely, it has integers 0 to p - 1. And my operation here is addition modulo p. You are given a generator, by the way, since the order of this cyclic group is a prime quantity because it has prime number of elements then, from the results that we know till now, every element of this set $\mathbb{Z}_p$, except the identity element 0, will be a generator.

So, 1 is a generator, 2 is a generator, 3 is a generator, p - 1 is also a generator. So, now let us see an instance of discrete log problem and how efficiently we can solve it. So, you are given the generator, you are given the random y and your goal is to come up with a unique exponent x, such that, g^x would have given you y, namely, you want to compute a discrete log of y.

By the way, the interpretation of g^x here, will be in the additive sense, we are not going to multiply g because our underlying operation here is addition. So, g^x should be interpreted as $x \times g$. So, imagine x is the discrete logarithm of y and that is the case then, y is nothing, but g added to itself modulo p, x number of times and our goal is to find out what exactly is x.

So, I know that y is nothing but x times g modulo p because g added to itself modulo p is equivalent to saying that, I multiply x with g and then take mod p. And my goal is to find out this unknown x. Now, it is easy to see that this unknown x is nothing but the product of y with the multiplicative inverse of g modulo p. So, this g inverse is now not the additive inverse. This is now the multiplicative inverse of g modulo p.

And now you might be wondering whether the multiplicative inverse of g modulo p exists or not. Indeed, it exists because the generator g is an element in the range 0 to p - 1 and it is co-prime to p. In fact, all the elements 0, 1, 2 up to p - 1 are co-prime to p because your p is a prime. So, the generator g is also co-prime to p. And then, since the generator is co-prime to p, we know that its multiplicative inverse modulo p exist which we can easily find out using Euclids algorithm.

So, now you can see that here, I do not need to do a brute force, I do not need to check whether x = 0 satisfies relation y = x times d modulo p or not, I do not need to check for x = 1, x = 2 and all the way x = p – 1. I do not need to do that because I know that, just by multiplying y with the multiplicative inverse of g, I can hit upon that right x. So, this is an instance of a cyclic group where D log is easy, very easy to solve.

Now, let us see another cyclic group where it is conjectured that discrete log problem is really difficult to solve for a random instance. So, consider the cyclic group $\mathbb{Z}_{p^*}$ where $\mathbb{Z}_{p^*}$, will have all the integers which are relatively prime to p and p itself is a prime number. So, if that is the case then $\mathbb{Z}_{p^*}$ will have all the numbers except 0 from $\mathbb{Z}_p$. And my operation is now multiplication modulo p in that group.

So, we know that this is a cyclic group. And, in fact, later we will prove it concretely that this group is indeed a cyclic group, but for the moment, you have to believe me that this group is indeed a cyclic group. So, now let us see a random instance of the D log problem in this group. So, you are given a generator, you are given a random value y and your goal is to compute a discrete log of y.

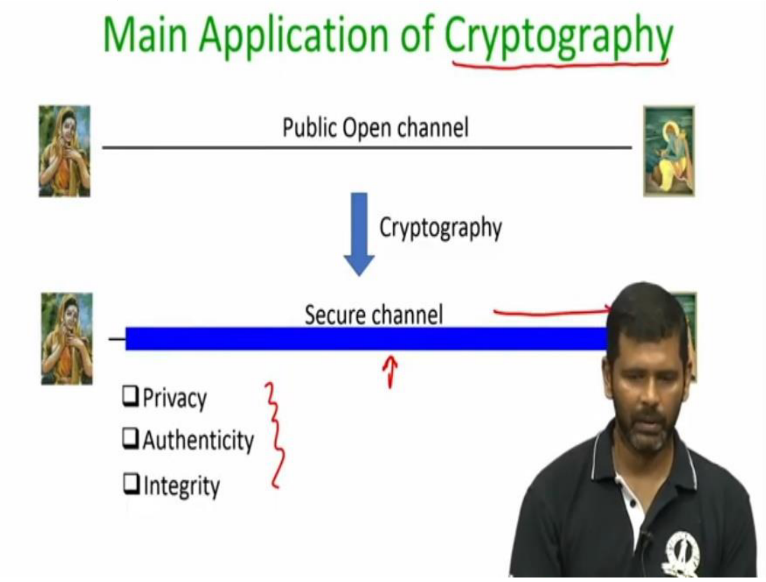
So, this should be $\mathbb{Z}_{p^*}$ because y is an element of your group and the group that we are right now considering is $\mathbb{Z}_{p^*}$. So, imagine $y = g^x mod\ p$. And it turns out that we do not have any better algorithm to compute x, other than naive brute force algorithm that we have discussed in the last slide. And this is because there is no pattern available because of the mod operation which I am performing. What do I mean by no pattern available?

So, by that, I mean that, if I keep on increasing the value of x and check the value of y where $y = g^x$ to the power x modulo p then it is not necessary, it is not the case that as your value of x increases, the value of y also increases. You compare this with the corresponding function where I do not do a mod p operation. Suppose my function $y = g^x$ then for such a function, I can confidently say that as the value of x increases, the value of y also increases.

But as soon as I redefine my y to be $g^x mod\ p$ then because of the mod p operation, it is no longer the case that, as the value of x increases, the value of y also increases, it will increase, decrease, increase, decrease, increase, decrease and there will be absolutely no pattern in which the values of y increases and then suddenly dips and then suddenly increases and so on.

So, there will be a complete chaos if you plot a chart or a graph between x and ys with respect to the fixed g. And it turns out that we cannot find out any pattern and hence, if my x is not known to you, if I do not give you the value of x and just give you the value of y and it will be very difficult in general to compute the value of x, if my group is a sufficiently large group.

(Refer Slide Time: 17:24)



So, now, let us see some applications of the discrete log problem in the context of cryptography. So, let me tell you something about cryptography. So, it is a mathematical science, and the main goal of the cryptography is to establish a secure communication channel between 2 entities say, Sita and Ram, who do not know anything about each other, they are meeting for the first time over the internet, and they want to talk over the internet by exchanging public messages.

And at the same time, they would like to ensure that, no one else should be able to find out what exactly they are communicating. So, main application of cryptography is that we would like to run some algorithm. And using those algorithms we would like Sita and Ram to exchange messages, so that, it should give them the effect of a secure channel, secure channel in the sense, it would look like as if Sita and Ram are doing conversation over very secure channel which provides 3 properties.

It should provide the privacy of the communication, namely it means that, even if a third party observes whatever communication is happening between Sita and Ram and even if the third party knows the protocol description, according to which Sita and Ram are doing the conversation, still that third party should not be able to figure out what exactly Sita and Ram actually are talking about that means, the actual contents of their messages.

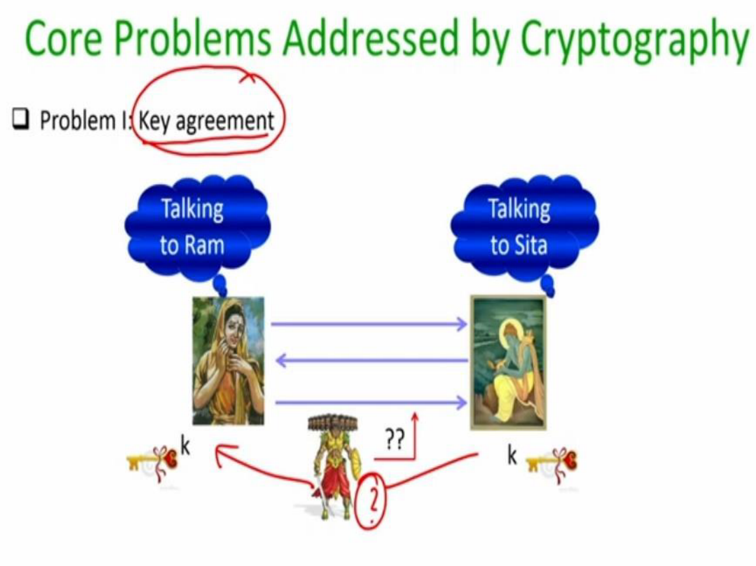
So that is a rough definition of privacy, we also need the authenticity property namely any message or any packet which is coming to Sita, it should have a proof that indeed it came from the person called Ram and in the same way any packet which goes to Ram, there should be a proof that or there should be a mechanism to verify that indeed that packet came from the person called Sita. So that is a rough definition of authenticity property.

And the third requirement of this secure channel is that of integrity. That means, if there is a third party which messes some of the bits or contents which are exchanged over the secure channel then it should be detected at the receiving end. So, through cryptography, we achieve all these 3 properties. So, basically cryptography gives you a set of algorithms, a set of protocols, according to which Sita can convert her messages in some format and communicate to Ram and vice versa.

So, there are lots of applications of cryptography. So, for example, if you are a user and if you are doing a net banking transaction then you are supposed to give your net banking password, at that time, you do not want your net banking password to be revealed to a third party, it should be securely communicated to the bank. So that is an application of cryptography.

In the same way, whenever you are buying something on the internet on Amazon, you are asked to enter your credit card details; again, you would like to exchange or send your credit card information in a secure way without any third party knowing about the exact details of your credit card information. So, again, cryptography is coming into picture. So, it turns out that cryptography is now used left and right in each and every application because slowly and slowly each and everything is becoming digital. So now, you might be wondering what sort of algorithms we use in cryptography?

(Refer Slide Time: 21:01)



So, now I will give you very simple algorithms based on number theory and abstract group algebra, abstract algebra that we have seen till now. So, the 2 core problems that are addressed by cryptography are the following. The first problem is that of key agreement. So, what exactly is the requirement in the key agreement problem? So, the setting is the following.

We have 2 entities Sita and Ram, who do not have any pre shared information that means, no secret question, secret date of birth, nothing. They are meeting for the first time and they are going to talk publicly over the internet. So, we need a protocol here according to which Sita and Ram should talk to each other, and the protocol description also will be publicly on that is also important.

It is not the case that, process by which Sita is going to decide her message is known to Ram beforehand and vice versa. Because I am assuming that they do not know anything beforehand. So that means, if at all they are going to use a protocol that will be publicly available. So, we need a publicly available protocol according to which Sita and Ram should talk to each other.

And at the end of the protocol, magically, both Sita and Ram should arrive at a common key k which is a binary string of some length. And the interesting property, the security property that I need from this key agreement protocol is that, if there is any third party who has monitored the communication between Sita and Ram and who knows the protocol description should not be able to figure out what exactly is the key k which Sita and Ram has output. It might look like an impossible task, but we will see soon, how exactly key agreement can be achieved?

(Refer Slide Time: 22:49)