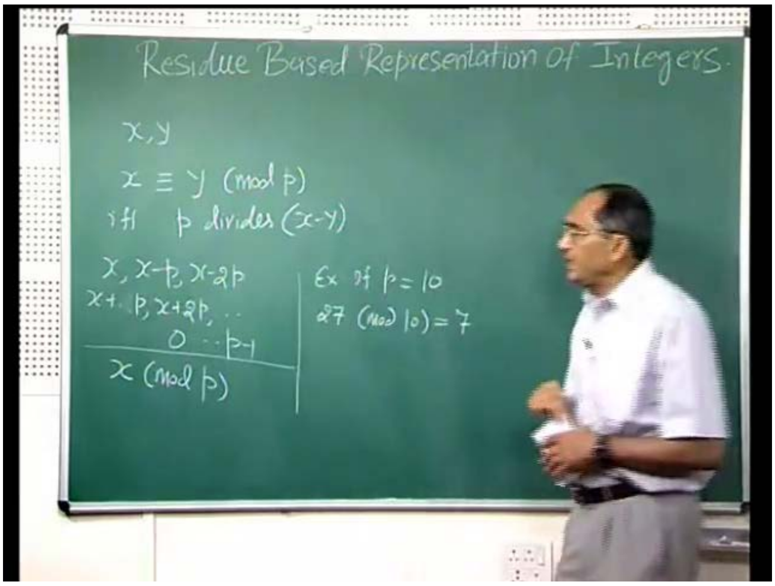
Computer Algorithms - 2
Prof. Dr. Shashank K. Mehta
Department of Computer Science and Engineering
Indian Institute of Technology, Kanpur

Lecture - 20
Chinese Remainder – I

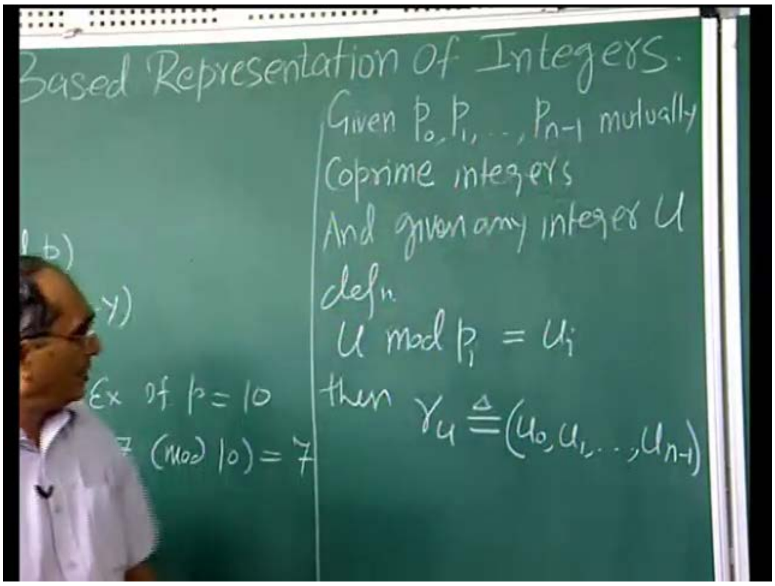
(Refer Slide Time: 00:26)



Today, we will discuss a representation for integers based on some residues. So, let us suppose we have 2 integers x and y, well we will only be dealing with non negative integers, then we say x is congruent to y modulo some other integer t, if and only if t divides x minus y. Now, therefore, x x minus p x minus 2 p, similarly x plus 2 p and so on, now x plus p rather x plus 2 p etcetera, are all congruent to each other modulo p. Hence there is always an integer in the range 0 to p minus 1, which is congruent to any given integer x.

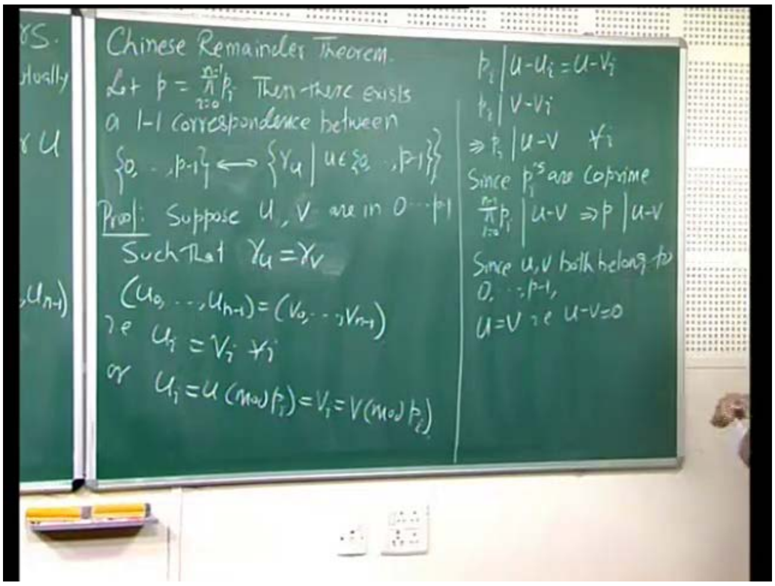
So, in today's discussion, we will denote such integer as x mod p. So, this is an abuse of notation and this denotes that integer, which is congruent to x and is in this range 0 to p minus 1; for example, if p is 10 then 27 mod p that is 10 is 7, so this is our notation. Now, today we will describe a representation for integers using these numbers, which are residue modulo p.

(Refer Slide Time: 02:35)



So, let us suppose, we have some n integers p 0 p 1 through p n minus 1 mutually, co prime, that is pick any 2 of these integers and their g c d is 1 given such. So, I should just say given and given any integer u integer u, we will define u mod p i as u i. So, if I compute the residue with respect to each p i for this integer u as u i then r u, which I am defining as a topple u naught u 1 u n minus 1. So, for any integer u, we define this topple, which is the residue of u with respect to each of these integers p.

(Refer Slide Time: 04:31)



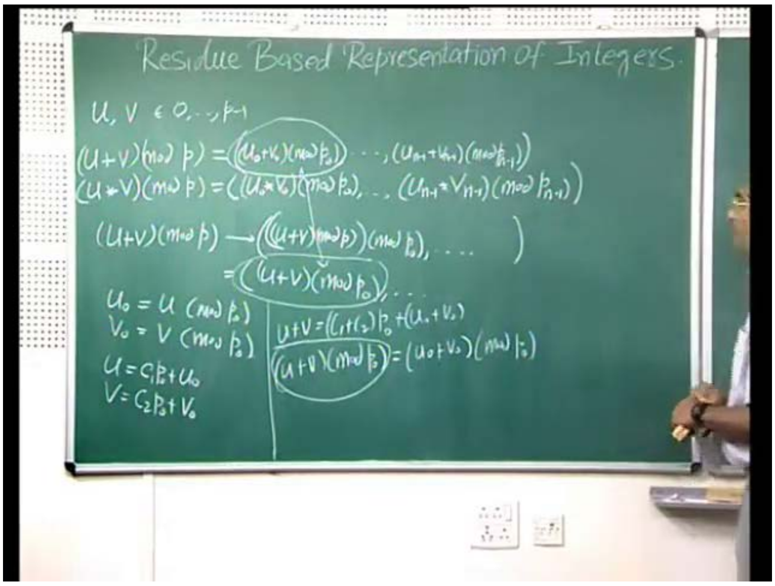
So, we describe the useful theorem called Chinese remaindering theorem, which says let me be the product of p i 0 2 n minus 1, then there exists a 1 to 1 correspondence between the set of integers and the topples r u such that u belongs to this corresponding set. In other for every integer in the range 0 to p minus 1, there is a unique n topple r u, which is given by the residues of that integer with respect to the n integers p 1 p 0 through p n minus 1.

To prove this well it is obvious that for any integer, there is a unique topples, because there is a unique residue with respect to h p i. Now, let us try to prove the converse suppose, u 1 and u 2 are in the range 0, through p minus 1, such that r of u 1 is same as r of u 2, in this case, we want to show that u 1 is actually equal to u 2. So, let us say that, we have let us use some other integers u and v u and v and we want to show and assume that, r u is equal to r v, we want to show that u is equal to v.

So, what we have is u 1 rather u 0 through u n minus 1 is equal to v 0, through v n minus 1, because of the fact that the 2 topples are equal, that is u i is v i for i or u i, which is u modulo p i is equal to v i, which is v modulo p i. Hence so t p i divides u minus u i, which is equal to u minus v i, p i also divides v minus v i, hence p i must divides the difference of the 2, which is u minus v.

This is true for every i, since p i's are co-prime the product of p i's 0 to n minus 1 also divide 0 minus e, which is same as p divide u minus v, but by our choice both u and v are in the range 0 to p minus 1. Since u and v both belong to the range 0 through p minus 1 u must be equal to v, that is u minus v should be 0. Which establishes the 1 to 1 correspondence between these topples and these integers. So, in some sense these topples represent, these integers and in certain computations, this representations can be very handy 2 binary operations namely addition and multiplication can be done, in this representation as follows.

(Refer Slide Time: 10:07)

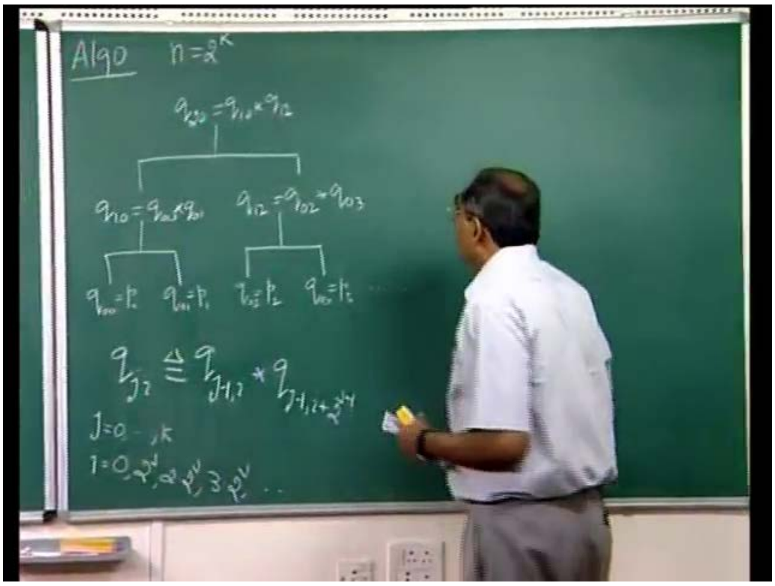


So, let us suppose, we have u and v in the rage 0, through p minus 1 then u plus v modulo p, which is the representation of the sum of these 2 or the number corresponding to u plus v in this range. This is equal to u 0 plus v 0 modulo p 0 u n minus 1 plus v n minus 1 modulo p n minus 1. What we have here is that the representation of u was u naught u 1 u 2 u 3 u n minus 1 and that of v was v naught 2 v 3 through v n minus 1, then all you do is you take the corresponding components add them up and compute the modulo with respect to the corresponding p and you get the representation of this number.

And the similar claim is true for multiplication v n minus 1 modulo p n minus 1 to see this let us say take 1 the first case u plus v modulo p as a representation namely, u plus v modulo p modulo p naught and so on. This is how, we will write the representation of u plus v modulo p, but this is equal to u plus v mod p naught and so on. The reason is that is p is a multiple of p naught 2.

Now, let us take u naught, which is nothing but u mod p naught and v naught is v mod p naught. So, u is some C 1 p naught plus u 0, v is C 2 p naught plus v 0 u plus v therefore, is C 1 plus C 2 p naught plus u 0 plus v 0, hence u plus v modulo p 0 is on this side, when you do modulo p 0, this vanishes and you are left with u 0 plus v 0 mod p naught. And this is exactly what, we are computing this quantity is same as this and therefore, these 2 are equal. So, this is true for any p i and similar argument holds for the multiplication. So, this is just telling us that, you can perform these operations in the new representation, now let us talk about how to compute efficiently, the new representation.

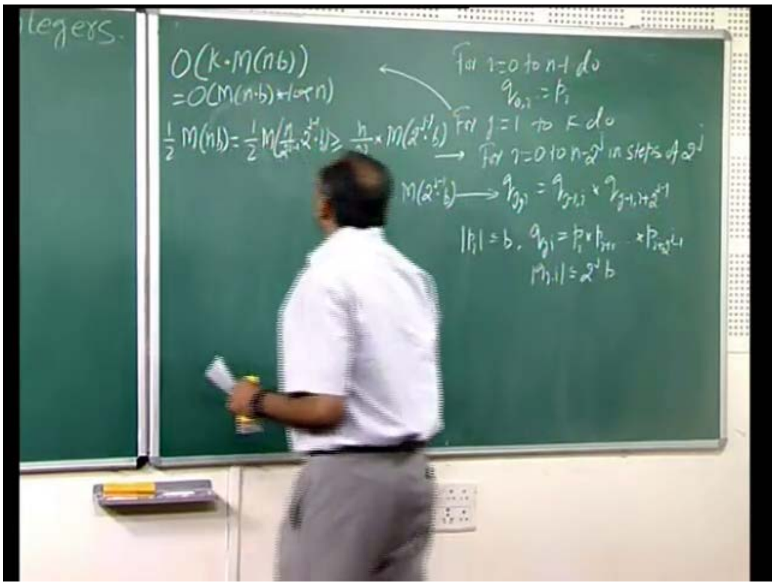
(Refer Slide Time: 15:08)



So, let us come to the algorithmic aspect of this computation, first of all let us define q 0 0 as p naught q 0 1 as p 1, q 0 2 as p 2, q 0 3 as p 3 and so on. To be able to perform the computation, we are going to do, we will assume that n is a power of 2, this goes on. So, if I multiply these 2, that I am going to denote as q 1 0 as q 0 0 times q 0 1 q 1 2 as q 0 2 times q 0 3 and so on, product of these will be called q 2 0 as q 1 0 times q 1 2.

So, we define in general q j i as q j i into q j i plus 2 to the power sorry, j minus 1 i q j minus 1 i into q j minus 1 i plus 2 power j minus 1. So, where j ranges from 0 through k and the range for i will be i will range from 0 to 2 power j, 2 power 2 times 2 power j, 3 times 2 power j and so on. So, how do I compute these well exactly the way, we have stated.

(Refer Slide Time: 18:09)

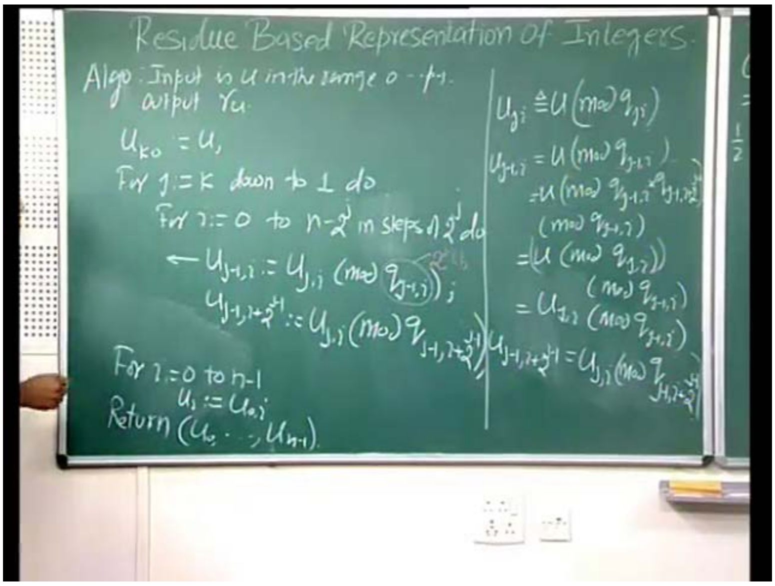


So, to compute the q i's, we will say 4 i equal to 0 2 n minus 1 q 0 i is p i for k equal to 1 to n do 4 i equal to 0 to n minus 2 power j in steps of 2 power j, which means 0, i equal to 0, i equal to 2 power j, i equal to 2, 2 times 2 power j, 3 times 2 power j so on, the last being n minus 2 power j remember 2 power j divides n. We write down the same relation q j i equal to q j minus 1 i times q, j minus 1 i plus 2 power j minus 1.

Now, that size of h p i is at most b and as q j i is same as p i into p i plus 1 all the way up to p i plus 2 power j minus 1. So, the size of q j i is at most 2 power j times b, this operation takes m 2 power j minus 1 times b, which indicates multiplication of 2 integers of size 2 power j minus 1 times b, that is the size of these 2 integers. The time it takes to complete this loop is 2 power n bar 2 power j into m 2 power j minus 1 times b, that because of the fact that m is version linear can be bounded by n by 2 into rather 1 by 2 into n by 2 power j minus 1 my mistake.

We will say 1 by 2 m n by 2 power j minus 1 into 2 power j minus 1 into b, which is equal 2 1 half m n b. Finally this loop, which runs k times, so the total complexities of the order of k times m n b, which is same as order m n b into log of m, which is k. Now let us try to compute the modular representation of any integer u in the range 0, through p minus 1.

(Refer Slide Time: 22:14)



So, let us write down the I will go the input is u in that range 0 through p minus 1 and output is suppose to be r u the topple residual residues with respect to each of the n integers. So, this time, we proceed from the in the opposite direction, let us define u let us separately first define it, here let us define u j i to be u modulo q j i.

So, what is it is relationship with u j minus 1 i's, so let us take a look at u j minus 1 i u j minus 1 from the definition is u mod q j minus 1, I which is same as u modulo q. Let us observe that modulo before, I can expand this modulo rather any number x, modulo y is equal to x modulo y times z and the whole thing modulo y. That is to say if, we compute the remainder of a with respect to the division by y times z and then take that remainder and again divide by y, you will get the same result as that by directly by y.

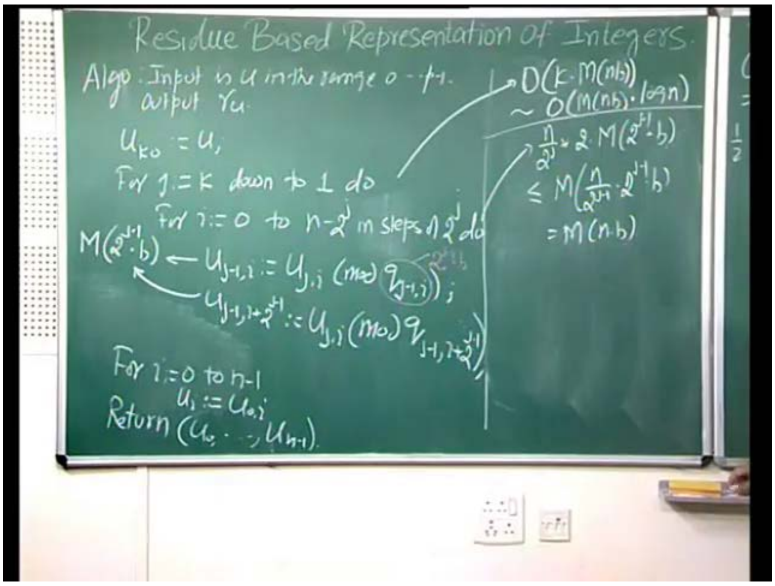
Now, if that is the case then we can use the following result, we can compute first of all u modulo j minus 1 i into q j minus 1 i plus 2 power j minus 1 and then we compute modulo q j minus 1 i. There this number is u modulo q j comma i and then we compute modulo j minus 1 i.

So, if we already have computed u j i then we can replace this expression by that value and compute modulo q j minus 1 i. Similarly, we can show that u j minus 1 i plus 2 power j minus 1 is also equal to u j i, but this time, we compute modulo q j minus 1 i plus 2 power j minus 1, then use these expressions to descend from u k 0 down to u 0 of various values.

So, we will begin with u k naught as u then 4 j going from k down to 1 and for i from 0 to n minus 2 power j, rather in steps of 2 power j, in steps of 2 power j do. And let us compute u j minus 1 i as u j i modulo q j minus 1 i and u j minus 1 i plus 2 power j minus 1, similarly is u j i mod q j minus 1 i plus 2 power j minus 1. So, this way finally, we will get u 0 i's, what is u 0 i u 0 i by our definition is u modulo q 0 i, which is same as u modulo p i, that we are calling u i. So, that is the these are the components of the representation.

So, we denote the u 0 i's as u i 4 i equal to 0 2 n sorry, n minus 1 u i is u 0 i. So, return u 0 through u n minus 1 well, the complexity of this process. Once again recall that these numbers have size at most let me just write this down is 2 power j minus 1 times b, 2 power j minus 1 times b therefore these numbers, cannot have size more than 2 power j minus 1 times p. This is 2 power j times b at most and you are dividing it by 2 power j minus 1 b.

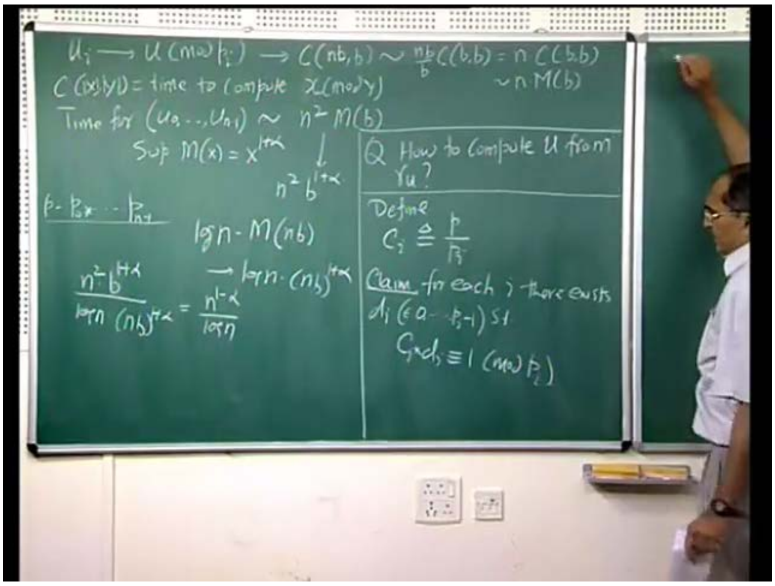
(Refer Slide Time: 30:19)



So, this modulo computation is a division of 2 power j size number 2 power j times b to size number by 2 power j minus 1 times b number, which from our earlier lectures, we have seen a same complexity as a multiplication of 2 power j minus 1 b size numbers. So, both of these involve this much computation and inner loop, the inner for loop runs n by 2 power j times.

So, we had n by 2 power j into 2 times m 2 power j minus 1 b time bound that again, because of verse then time complexity of multiplication has to be n by 2 power j minus 1 into 2 power j minus 1 b, that is m times b, once again the outer loop runs k times. So, the time complexity of this will be k. So, total time complexity is a times m and b, which is same as order m n b log of m, which is the same time complexity, it takes to compute all the q j i's, hence the entire process computation of q's as well as the representation of u takes this time. None it is compared their same computation, if it is done in a direct fashion.

(Refer Slide Time: 32:30)



That is if, we compute each u i, let us say u i, directly by computing u modulo p i then how much time it takes. Let us use a notations C of comma y as the time to compute x modulo rather x, x modulo y, for us u is a number in the range some 0 to p minus 1, size of p can be at most, the size of p note that, it is p 0 times p 1 times p 2 up to p n minus 1. So, this is of the size n time's b. So, this process will take C of if I write down the sizes of the 2 numbers, we are computing the modulo of a number of size n times, we are dividing this number by a number of size b. So, maybe I should say the sizes, this is a number in the range 0 to p minus 1, hence in the worst case it could of size n times b.

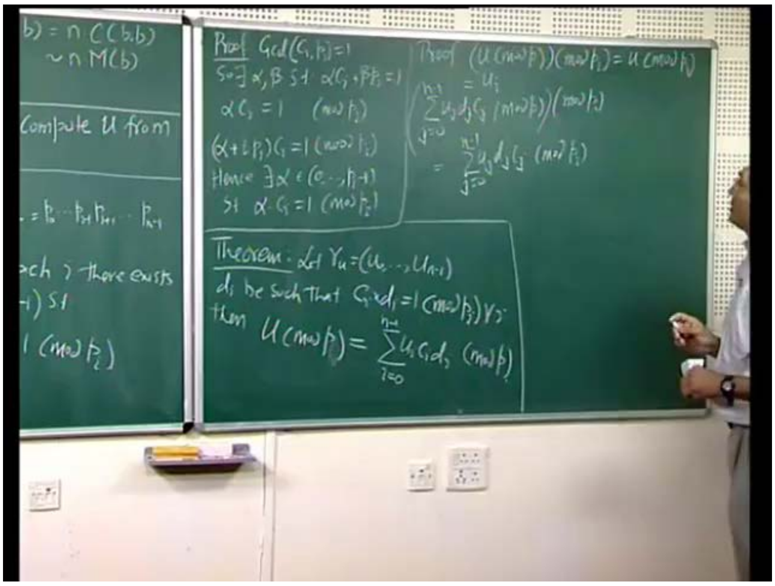
Now, if we are dividing a number by of size n b by number by b. So, it is a reasonable approximation that this is going to take about n b by b C b comma b that, because you have to divide in a n times larger number in size compare to the number or the divisor.

So, this is about n C b comma b, b comma b can be approximated as n b. So, we are taking this much time to compute 1 component. So, the entire representation the time, for the entire representation will be well approximately n square m b. Now to compare with our algorithms time let us, suppose n of x is x power 1 plus alpha, we know that it is version linear. So, let us say it is a about x to the power 1 plus alpha, then this thing turns out to be n square times b to power 1 plus alpha.

As compare to that the time, we have taken in the algorithm is n sorry, log of n times m of n b, which is log of n times n b power 1 plus alpha. So, if you take the ratio the speed up will be n square times b time b power 1 plus alpha divided by log of n times m b power 1 plus alpha, which is m to the power 1 minus alpha divided by log n. So, it shows that there, is a significant improvement as long as alpha is smaller than 1, which we know it is. So, now let us try and look at the problem of reverse computation. So, question is how to compute u from r u, so far, we have been showing how to go from u to r u, but how to go backwards. So, for that again there is a nice result, which allows you to compute, you from r u.

First of all let us try and define a few things, let us define C i as p divided by p i, now we make a claim that, for each i there exists a d i in the range belonging to the range is 0 to p minus p i minus 1. Such that C i into d i is 1 congruent modulo p i, in other words d i is reciprocal of C i modulo p i. So, to show this, let us observe that C i, which is a product of p 0 through p i minus 1 p i plus 1 all the way to p n minus 1.

(Refer Slide Time: 39:36)



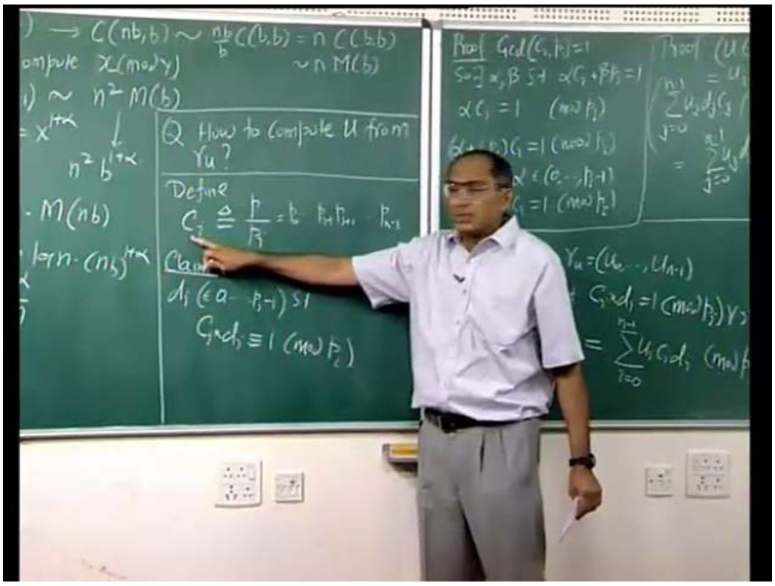
So, this is co-prime with respect to p i, because each of these is co-prime. So, G c d of C i and p i is 1. So, there exists alpha and beta such that they could be positive or negative, such that alpha C i plus beta p i is 1. If you take modulo p of this equation on both sides, what you get is alpha C i is 1 mod p i, because this vanishes, when you compute the residue with respect to p i.

Hence alpha is the number that, we are looking for, but observe that, if alpha does this then alpha plus any t times p i into C i is also 1 mod p i, hence any number here, can work as long as it is congruent to alpha with respect to p i. Hence there exists in alpha in the range 0 to p i minus 1, such that alpha times C i is 1 mod p i. So, this establishes the existence of this alpha is our d i.

Then we have the following theorem let r u b u 0 through u n minus 1 b i b, such that C i into d i is 1 mod p i for whole i. Suppose, we have given these then u mod p i the residue of u or we if you assume that u sorry, mod p not p i, if we assume that u is already in the range 0 to p minus 1 then this is same as u is equal to u i c i d i mod p. And to show the correctness of this claim, which is very easy let us take the modulo p i on both sides, so u mod p mod p i, which is the left hand side now.

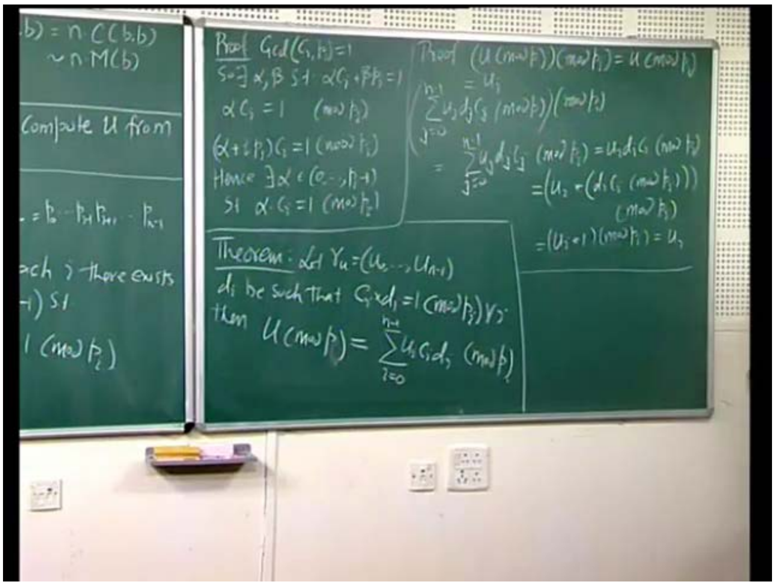
Note that p i is a factor in p, so this is same as u mod p i. So, the left hand side is nothing but u i well, which is correct, because after all this is what, we expect when, we compute the modulo p i on the left hand side. If this is indeed equal to this then we should also find that, the modulo p i of this sum is u i. For the right hand side, if I take the sum i equal to 0 to n minus 1 u i d i C i mod p mod p i, once again this is same as sum 0 to n minus 1 mod p i note that. I should change my index let us use some other index call it j.

(Refer Slide Time: 45:07)



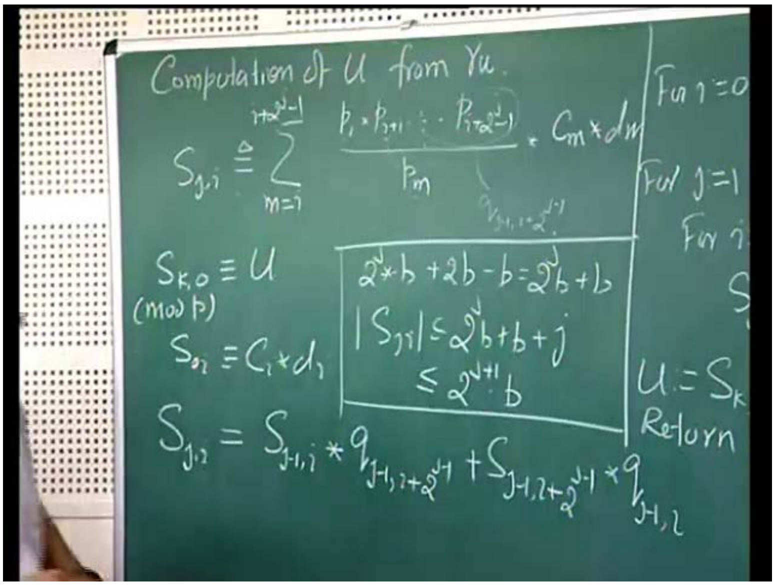
Because, this is a dummy index and we are computing modulo p i, every C j is divisible by p i except C i, that is how, we have defined our note that C i is all the product of all the p's, other than p i, hence it is divisible by every p j's other than p i.

(Refer Slide Time: 45:19)



So, C j mod p i is 0 for all j other than j equal to i. So, this reduces to u i d i C i mod p i other terms have vanished, hence this can be written as u i times d i c i mod p i, whole thing modulo p i, but this is 1. So, this is nothing but u i into 1 mod p i, but that is equal to u i. So, indeed this number has the same residue with respect to p i as u. So, from the uniqueness, we know that, this and this number are same modulo p. Now, finally, let us go ahead and compute, the integer u from components u i's.

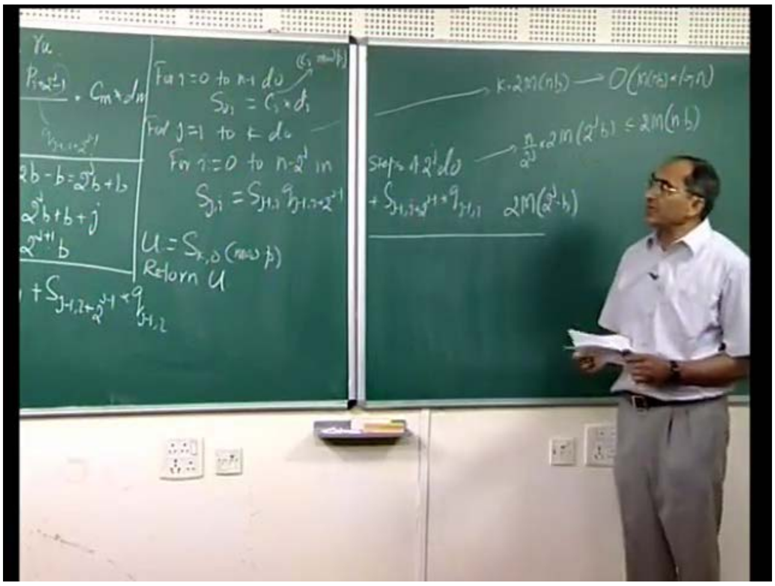
(Refer Slide Time: 46:38)



So, now to compute u from the residues, let us define S j i sum p i into p i plus 1 on the way to p i plus 2 power j minus 1 divided by m into C n into d n, where m starts from i n goes up to i plus 2 power j minus 1. If we define it, this way then S k 0 is precisely u provided v compute. So, let us assume that, we do modulo p S k 0, modulo p is u, the values of S 0 i's are precisely C i into d i. So, let us see the relationship, how the recurrence relation can be set on this, let us define, we will let us, we have this we have this h j i as defined here, we can now split this in the following way, that S j minus 1 i into q j minus 1 i plus 2 power j minus 1.

Note that, if m is less than i plus 2 power j minus 1, that is to say it is in this half, m is in this half then this whole thing will be nothing but q j minus 1, i plus 2 power j minus 1. So, we can take that chapter out from all those sums, where m is in the lower half and for higher half, we can take the terms out of this in the similar fashion and that is what it gives you. So, we get s j minus 1, i plus 2 power j minus 1 times q j minus 1 i. So, this allows us to actually, compute the value of S for higher values of j.

(Refer Slide Time: 49:51)



So, once again, we will set up a program starting with the lower level 4 i equal to 0 to n minus 1 S 0 i is C i into d i, we assume that d i has been pre computed. For j equal to 1 to k do 4 i equal to 0 to n minus 2 power j, in steps of 2 power j do S j i S j minus 1 i q j minus 1 i plus 2 power j minus 1 plus S j minus 1 i plus 2 power j minus 1 times q j minus 1 i and u is nothing but s k 0 mod p and return u the sizes note that. We can instead of using the C i's as it is, we can replace them in the range their modulo. So, we can replace them by C i mod p i, we can replace, we already have decided to use a number here, in the range 0 to p i minus 1.

So, the sizes of let us take a look at this S j i here, S j i has the size individual turn here has got 2 power j into b bits in the numerator, these also have at most b bits minus b bit. So, each of these is 2 power j b plus b bit size at most this is the maximum an individual term can have and we are adding up 2 power j such terms. So, adding 2 terms adds the size by 1 bit.

So, the total size of S j i is at most 2 power j b plus b plus j after adding up 2 power j terms, this is the maximum number of bits you can have, which we can comfortably assume to be plus 1 times b, this is the maximum size. So, this computation involves multiplying a 2 power j plus 1 b bit with the number with a 2 power sorry, 2 power j times b bit number with the 2 power j minus 1 size b times b bit number.

So, let us suppose, this involves 2 times m 2 power j into b multiplications, hence this will take n by 2 power j into 2 times m 2 power j times b, which is in our visual argument, we have shown that this is at most n sorry, 2 times m n times b bits. And this is k times 2 times m n b, once again the time complexity of this is m n b into log of n, which is k once again is the same time complexity it takes. So, this is all we have it takes the same amount of time to prove back and forth between the number and it is residual representation. In the next class, we will show that the same ideas apply to polynomials.