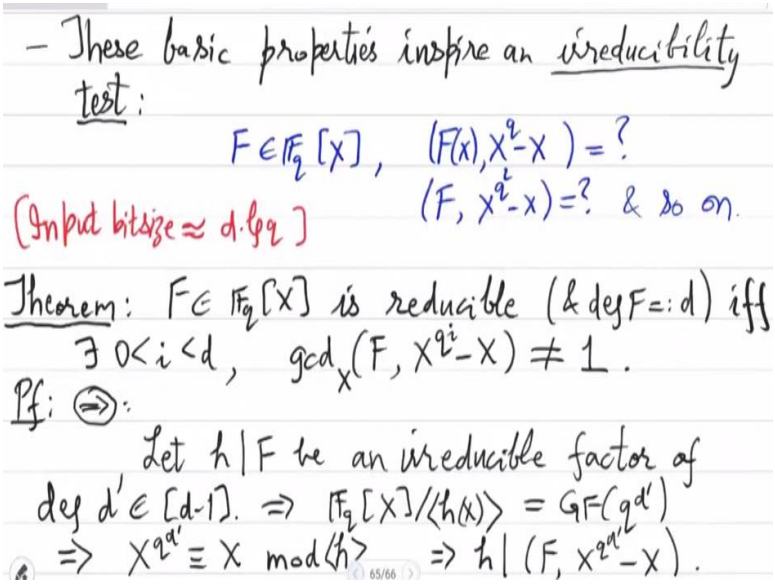
Computational Number Theory and Algebra
Prof. Nitin Saxena
Department of Computer Science and Engineering
Indian Institute of Technology - Kanpur

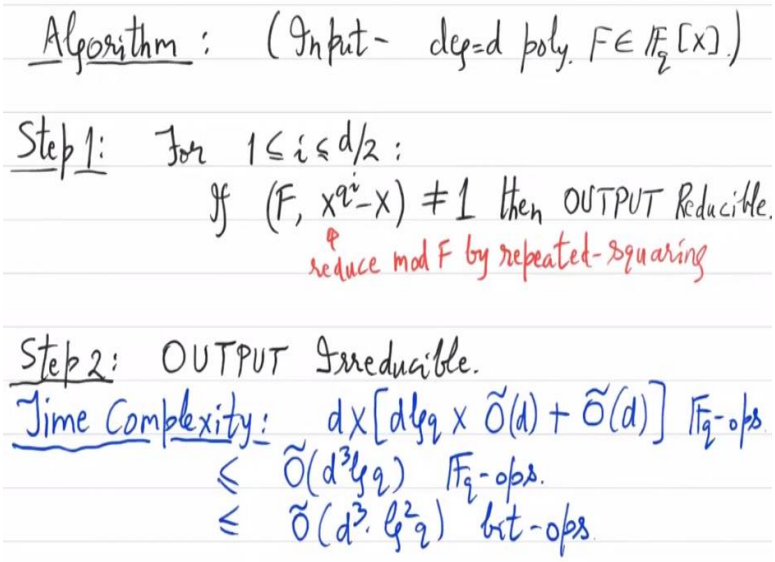
Lecture – 10
Equi-Degree Factorization and Idea of Berlekamp's Algorithm

(Refer Slide Time: 00:15)



Okay, so last time we started the topic of checking whether a given polynomial big F, look at the theorem, it has coefficients from a finite field. The finite field is also given to you and the polynomial is given to you in terms of coefficients and monomials, degrees d and ultimately we want to find factors of F, but first for that we have to check whether it is reducible or irreducible. So, we show that it is reducible if and only if one of these gcd's is not 1, can you give a proof of this?

(Refer Slide Time: 01:02)

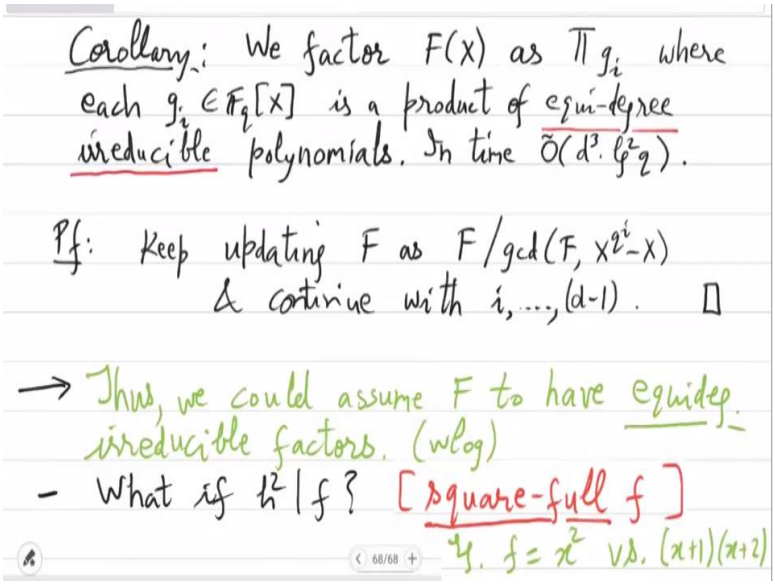


And based on that there is this straightforward algorithm that you go over all i's taking gcd with respect to this x variable of F with X to the q to the i – X, q is very large, q is given to you even if it is a prime it may be 100 bit or 1000 bit prime. So, its value is actually 2 raised to 1000. So, this X to the q computation is or the degree of X to the q – X polynomial is huge. It is something like 2 raised to 1000 degree.

So, you cannot store it if you wanted to store all the coefficients, but thankfully here the coefficients are mostly 0, it is only X to the q to the i – X. You just have to compute one monomial and that you will do by repeated squaring mod F. So, you only need its value modulo F because you only want to compute the gcd. So if you find the sum i such that the gcd is not 1, then you will output that F is reducible.

Otherwise, if you have checked for all i, then you will deduce irreducibility. So, this is essentially cubic time algorithm. So, all the way down to bit operations, this is a very efficient fast algorithm. In fact, this is the only algorithm you can think of for irreducibility testing.

(Refer Slide Time: 02:37)



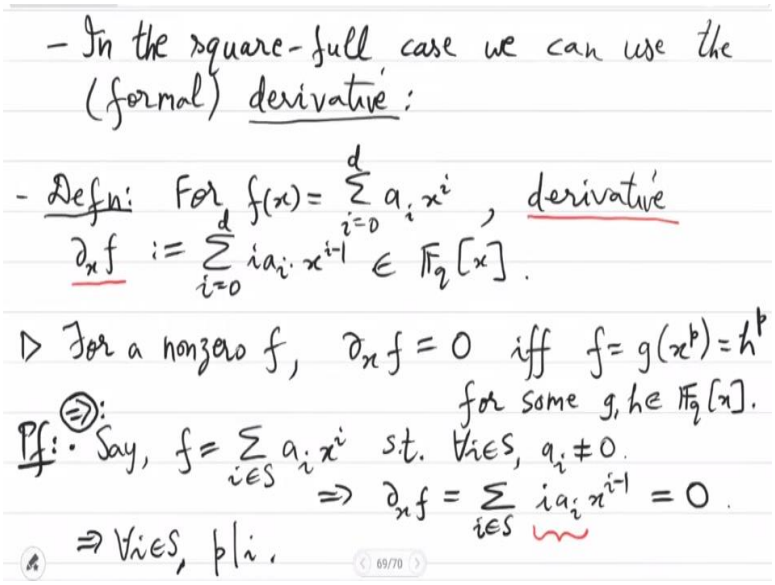
And this gives you more, so this actually gives you partial factorization of F. so, in case your input polynomial has 2 different degree irreducible factors, they will be separated okay. So, if you have a degree if big F has a quadratic irreducible factor and has a cubic irreducible factor, then for i = 2 the quadratic will come out okay. So, they will be separated, so, you will actually get what is called equi-degree factorization.

So from now on, we can assume that any irreducible factor of big F has the same degree. So, either a big F itself is irreducible or it has many reducible factors all of the same degree. This we can assume without loss of generality now so to speak okay because this algorithm, this is you have a fast algorithm to get to this point. One more property we can easily resolve is what there is an irreducible factor who square divides F okay, so that is called square full case.

So, what if h square divides f. So, that that is called square-full f and the opposite of this is a square free. So if all the irreducible factors of F they are distinct or they are co prime, we call F to be square free, otherwise we call F to be square full. So for example F = x square this is square full. On the other hand x + 1 times x + 2, this is square free, okay. So, x + 1 times x + 2 is the roots 1 and 2 will be distinct, even more 2 they will be distinct.

While in the case of x square, there are two roots, but both are 0, right, so square-full versus square free. This can we achieve square freeness is the question by a fast algorithm.

(Refer Slide Time: 05:08)



So, in the square-full case, we can use the formal derivative. So, what is derivative of a polynomial? So for fx sigma ai z to the i d degree. This has only one variable, so the derivative is denoted by dou x f and what is that? Yeah, so, the derivative in complex analysis is defined by some limits, etc., right, but what is the final result? The final result is that x to the i derivative is just i times x to the -1, right.

So, that is a syntactic expression. So bypassing all the analysis, we just define that to be derivative okay. So, since over finite field, we do not have analysis available, so we do not want to use limits, etc. We just say that this polynomial where the coefficient of x to the i-1 or so hence the coefficient of x to the i is i+1 times ai+1 okay. So, it is slightly shifted and it is scaled up by i.

This polynomial is somehow related to f and will turn out to be useful and this is also a polynomial in the same field. So, when is this derivative 0? What does it mean if this formal derivative of f vanishes absolutely, sorry f is what? Yeah, yeah, sure. If f is constant, then obviously derivative vanishes that happens in analysis, but here you are over a finite field fq characteristic is p. So, what are the other ways of derivative vanishing?

Exactly so and i will be a multiple of p means that the monomial is a pf power. So, derivative vanishes if and only if f is some function of x to the p and you have learned before that, okay this is slightly different, for some g and h okay. So, one side is clear, if f is a function of x to the p when you apply the derivative operator since every monomial p divides i the derivative will vanish. So, every coefficient there this i = 0 to d.

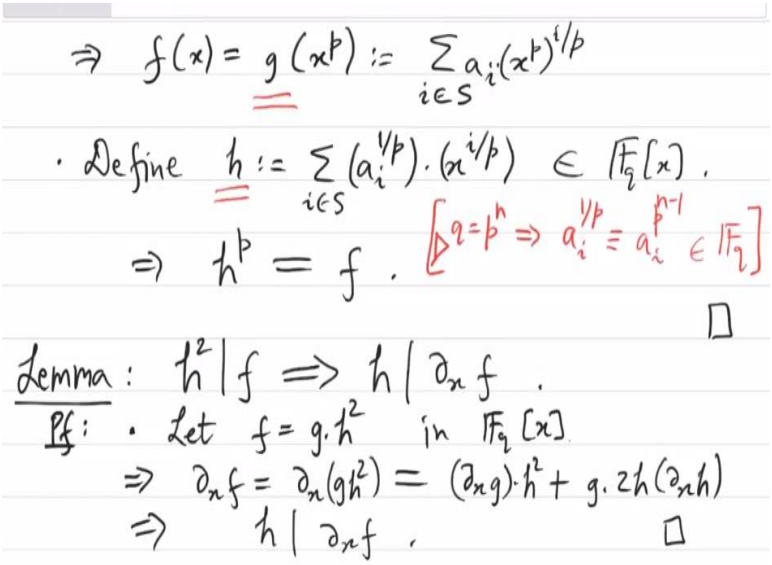
Yeah it is the same thing i = 0 term will not contribute, it is in bijection with the representation of f. So, every i is a multiple of p's and hence i times ai will vanish okay. So, one side is clear. The other side is similar proof. So, let us do that. So, let us say the support is s. So, none of these ai's are 0 to begin with, but when you differentiate what you will get is i ai x to the i -1 and this suppose is 0 okay.

So, we are looking at the forward direction. So, we are assuming that the derivative is 0, which means that look at this part. So, this ai was nonzero in Fq, but i times ai is 0 in Fq which means that i is 0 but i is a number. So, which means that p divides i. So, you get that every i is divisible by p, is that clear? So, hence f automatically defines g such that f is g of x to the p.

"Professor – student conversation starts." Sorry. And there will be **((11:03))** Yeah, but we are assuming that ai is nonzero. We are looking at the coefficients that actually appear that is the definition of s, s is the support of f. So, all these ai's by definition are nonzero. **"Professor – student conversation ends."** I mean the only other case is when f is absolutely 0, but then obviously 0 is a function of x to the p.

In all other cases, they will be the support s and here i will be forced to be divisible by p. So that gives you the definition of g.

(Refer Slide Time: 11:40)



So, which is actually, it is just this x to the p raised to i by p and how do you get h? So, you

can define h to be this ai to the 1 by p okay so for i in s. So, clearly h raised to p will be equal to ai x to the i which is then f. Now, so g clearly is a polynomial over Fq, where is h? Where are the coefficients of h? So, which is the same question as where is ai to the 1 over p. Yeah, so why if a is in Fq, why is ai to the power 1 over p is also in Fq?

Yes, so you have to use those properties that is the small exercise. So, I think we were assuming q to be p raised n. So, then ai to the 1 over p is actually the same as ai to the p to the n-1, right. This is coming from the Frobenius congruence or identity. So for every a to the p to the n, which is a to the q is a. So, in other words, a to the 1 by p is actually a to the p to the n-1 which obviously is an Fq.

"**Professor – student conversation starts.**" To make a to the 1 by p. Yes. It is unique. **"Professor – student conversation ends."** So, a to the 1 by p is a root of y to the p-1, right, y to the p-1 if it has a root, all of them are the same because of characteristic p. So, that shows that h is actually in Fq x. Okay, so we have defined g and h as required. Any questions? So, we just showed that if the derivative of a vanishes, then f is actually a function of x to the p, a polynomial in x to the p which is also a pth power okay.

So derivative vanishes only for pth powers. In no other case can the derivative vanish. So the pth power is actually, yep, let us delete that for a while. Let us look at the effect of square fullness next. So, see h square divides f okay f is square full, then what can you say about the derivative? I want to make a statement about the derivative. If h square divides f, h will divide the derivative okay.

So, this is a very useful property computationally because this gives you immediately an algorithm to test square fullness or square freeness, right. So, if there is an irreducible factor whose square divides f, then how can you discover it? It will divide both f and the derivative. So, you just take the gcd, both of them, I mean the derivative is very easy to compute and then the gcd is easy to compute, so when you take the gcd square fullness will be discovered.

When you compute the derivative, you may get it to be 0, right that will also happen, but if the derivative is 0, then you already know that f is a power of p. So, that is also an evidence of square fullness, right. So, either you see the derivative 0 which means something very strong about f or when you take the gcd of this nonzero derivative with f you get something

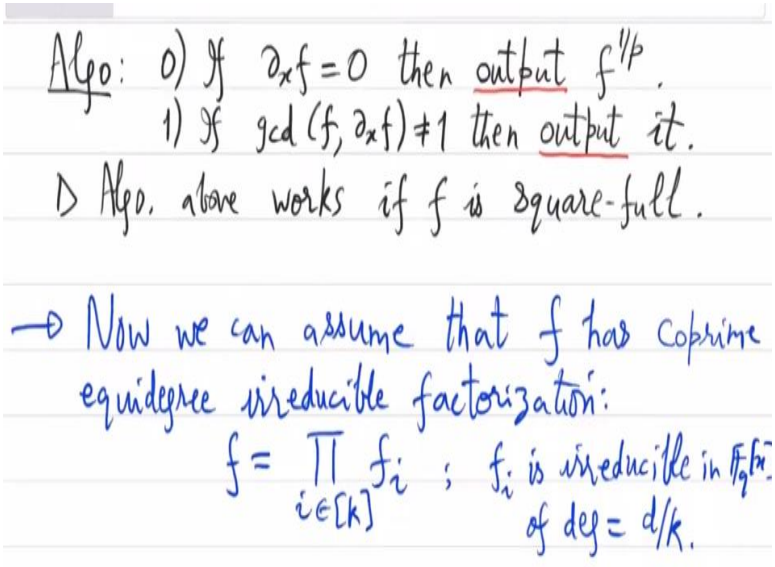
which also is an evidence of square fullness. So, either way you learn something using these two lemmas.

"Professor – student conversation starts." But in the second case we cannot calculate the if, derivative is zero, yes. No, then you calculate more, then you know that f is actually power of p, so you can remove this extra p and the degree reduces. You actually go to a much smaller case and then you will study that recursively. I think we should know what the is power of a **((17:47))** take the whole thing to the power. Yeah, that is in the proof. **"Professor – student conversation ends."**

This is something that you must have seen in school when you learnt derivative operator. So, let f be g times h square factorizes this way in F q x. So, if you apply the derivative operator, how do you expand derivative of product? So, there is this derivative product rule, right? So, you first differentiate g and then you differentiate h square and you see that each divide this, so that is it. So, this is just the product rule of derivative.

So, recall that you have seen in analysis that if you find a root of the derivative, which is also a root of f, you find a common root of f and the derivative, right. So, what is the speciality of this root, it is a repeated root. In other words, it has multiplicity at least two right. So, this is what the lemma is saying also, but the lemma here is completely syntactic or algebraic okay. There is really no analysis in this okay.

(Refer Slide Time: 19:44)



So, now we can assume square freeness because you can use this algorithm one step

algorithm with just outputs gcd of F with the derivative. So, this algorithm works if f is square-full. The case of derivative being 0 also I should somehow capture here, let me change this. If gcd of f with a derivative is nontrivial. Okay, so let me capture whatever we have learned till now to do the derivative in these two steps.

So, the first step you just check whether the derivative is 0. If it is 0, then you know that f is a pth power and you saw the definition of h before f equal to h to the p, so you just have to output that h okay, this is a trivial computation The other thing you can try is compute the gcd of f and its nonzero derivative, if it comes out to be non one, then you are sure that this is a nontrivial factor right because its degree cannot be equal to degree of f, derivative has a degree one less, right.

So, gcd either it has degrees 0 which is equal to 1 or it has degree between 1 and d-1 so it is a nontrivial factor. So, you output it okay. So, one of these outputs will be successful as long as f was square-full okay. If f is square-free, then both these steps will fail, is that clear? So, that is the computational content of whatever we discussed till now.

"Professor – student conversations starts." In the first step, we need to carry out the gcd again right a to the power 1 by and gcd of f we have to find it out again too. Sorry, you just output, just reduce f to the 1 by p, it is a factor of f, it is a repeated factor of f. It repeats p times. Irreducible we cannot claim. No, there is no such claim. Success just means output a nontrivial factor, that is all. **"Professor – student conversation ends."**

So, after all this work, we are at this point. So, now we can assume that f has coprime equidegree factorization. So, when you look at the factorization of f into irreducible, each of the irreducible has multiplicity one, they are all distinct and in fact coprime , they all have the same degree okay. So, after the previous two algorithms, you get to this point **"Professor – student conversation starts."** Sir, the algorithm is what irreducible value, did that also give us, equidegree.

It gives equidegree and they did also give us coprime as well because all the factors will be multiplicity one. Yes, so I think that there you have to take gcd with x to the q to the i minus x many times because it reduced one by one. No, my point is if we take it only one, then you get each factor of multiplicity one and then no need to take it multiple times then. Yeah that

is true. So this probably is, yeah this may be an extra step then, it is probably not needed.

Yeah just first algorithm you should take gcd. **"Professor – student conversation ends."** When you take gcd with x to the q to the i minus x that already is a product of coprime, equidegree, irreducible So, you start getting these things separated out that is true. So, we can write f, I mean we do not know the factorization, but we have the promise that it looks like this. **"Professor – student conversation starts."** Whole f was g square directly.

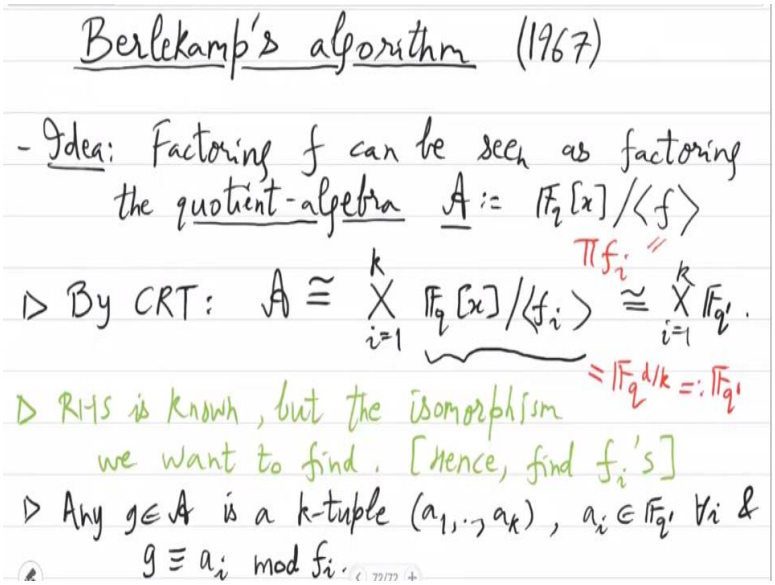
The whole f was directly g square. See if it was h square, when you take gcd of h square with proper x to the q to the i minus x, you will get out h. So, see f has k such factors and fi is irreducible in the finite in this polynomial ring of degree how much, they all have equal degree and f has degree d, so it is d by k okay. So, now, we are in this setting that so you can assume that whatever polynomial is given to you by either of these algorithms.

When you use them judiciously you will get to the point where fa has k irreducible factors each of degree d by k and they are unknown, but k is known right, why is k known? **"Professor – Student conversation starts."** How do you know how many factors there are? We know the overall degree and **((26:49))** of each values so. Yeah, how do you know the degree of each factor? We must calculate **((26:56))** gcd.

So when you take gcd with this x to the q to the whatever m – x right, so that m that exponent m is the degree of irreducible factors. So, you know the degree of fi, hence you know d by k from which you know k. **"Professor – Student conversation ends."** Yeah, so, this is the setting. So, any questions at this point? So, this sequence of algorithms is called preprocessing for factorization okay.

This already needs some development, but it is merely preprocessing because still you have not really found a way to completely factorize okay, this is still a bad case. So, at this point, now we will develop algorithms in a different way to further factorize and ultimately completely factorize in polynomial time. So, first algorithm in this area was due to Berlekamp.

(Refer Slide Time: 28:10)



So**,** Berlekamp's algorithm. This will not be the full solution, but it is still applied, it is still used in many real situations also in practice. So, remember this blue part, from now on we will just implicitly assume this okay, without saying we will assume that input polynomial F looks like this. So, the idea here is factoring f can be seen as factoring the quotient ring. So this quotient algebra is a Fq x mod f.

And on this you to understand this quotient algebra apply Chinese remainder okay what do you get? So by Chinese Remainder Theorem, this algebra A and remember we are assuming that blue assumption which is that f is a product of fi. So, the quotient algebra factorizes as direct product of Fq x by one factor which is fi, right. So, this algebra it factorizes into this k again quotient algebras and what can you say about this quotient algebra Fq x mod fi.

Sorry yeah fi is irreducible, so this is a field and what type of a field? It is again a finite field. So, this is Fq fi has degree d y by k. So, we will call it Fq prime okay. So, this is nothing but k copies of the finite field Fq prime okay. So, this structure is known, but what is unknown is the isomorphism okay. The structure is known because you know k and you know q prime, so you exactly know what the structure of the algebra is.

But what is missing is this Chinese reminder isomorphism that is not known. Well if that was known, then you would have been able to recover fi's, actual factors of f, right. So, this is the way we see factorization of f. We think of this as an unknown isomorphism. RHS is known, but the isomorphism we want to find hence find, hence find fi's. In fact, let me clarify this a bit more. So, when I say structure I mean this.

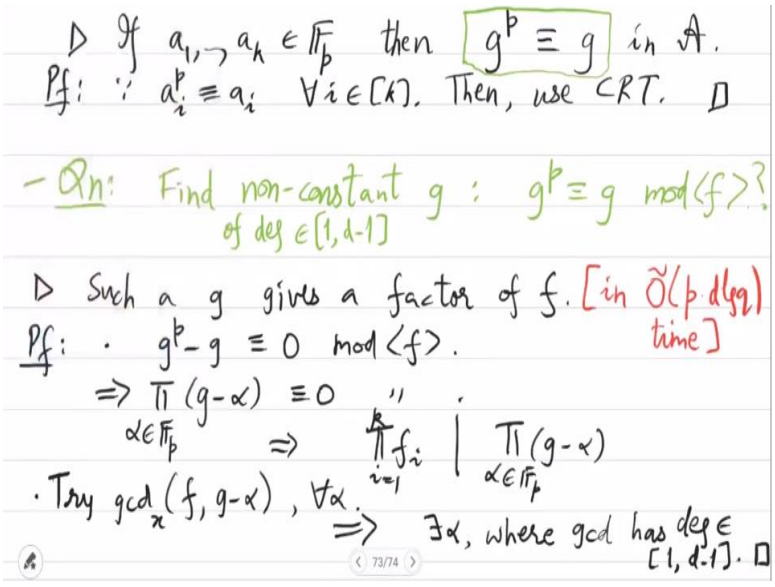
So, this last thing direct product of Fq prime k many this is the only thing which is known okay, the previous isomorphisms are not know. You do not know how to map Fq prime to that thing Fq x mod fi because you do not fi okay. So, when I say RHS is known it is only the last thing. So, with the abstract finite field, you know the structure, but you do not know how it is being realized in the current input situation that you have to find.

Any questions at this point? So, any ideas how will you find, using this quotient algebra, how will you find a factor? Find f1, f2, dot dot fk. Okay, so one more thing that we can deduce from this structure is any element g of A is a k-tuple, a1 one to ak, right, So, you pick any element in this quotient ring Fq x mod f, let us call it g and in this unknown isomorphism, this g Chinese remaindering will map to a k-tuple.

Each ai is in this finite field Fq prime and this element gx is the same as ai mod fi. Okay, so this is again just invoking Chinese remaindering. So Chinese remaindering takes an element of the LHS a let us call it g and just look at the remainder , so g mod f1, g mod f2, g mod fk will give you these coordinates of the kth that is how the isomorphism was, but since we do not know fi, we will never know ai, right.

We can only go there as we go over g in a, we know that these ai's exist, but right now we cannot compute them, but it is a strong property that they are in a finite field, right. These are known elements ai in the finite field fq prime. We will try to play with that.

(Refer Slide Time 36:21)



So, the first property is what if you focus on those possibilities of a1 to ak which are in Fp, so basically just numbers 0 to p-1. Then what can you say about g? So what are the g's whose k-tuples are just numbers, elements 0 to p-1? No, that is a mistake. Yeah, no g is not constant, g will still be, g is constant only when all the ai's are the same constant because a constant mod fi will be the same constant, right.

But as soon as you take a1, one a2 different constants like even if you take 0, 1 that does not correspond to a constant g, it is actually a nonconstant g. No, so I want to make it specific what can you say about g to the p? I just want this property. For these ai's as tuples as coordinates in the tuple, what can you say about g to the p? It is same as g right? So, that is a very useful property that is what Berlekamp algorithm is based on.

So, the reason for this is when you do Chinese remaindering, then you will basically be looking at ai to the p, ai to the p is a and this is true for every coordinate. So if it is true for every coordinate, it is true for the whole tuple. So, the whole tuple raised to p is equal to itself, which means that g is to p is equal to itself. Is that clear? And then use Chinese remaindering. Chinese remaindering theorem is an isomorphism.

So, when you get this relationship between ai to the p or this tupled raised to p equal to itself by isomorphism, the same thing is satisfied by g okay and yeah so for various reasons which may not be clear right now, but they will soon get clear this is the congruence which we will work with okay. This g raised to p equal to g, we want to find a g first that satisfies this congruence. So, how easy is it to find those g's? Well g = 0 satisfies this, g equal to any constant satisfies this.

So g constant, finding them is trivial, but they will not be of any help to us, we actually want to find a nonconstant g which satisfies this okay and finding that is hard. At this point, you do not know any nonconstant g that will satisfy g to the p equal to g mod f, right. So that is the first question. That in fact will be the most important question. So what we will do next will be solving that problem.

So, basically find non-constant g such that g to the p is g mod f, so can you find this? That is the algorithmic questions. Now once we have found it, how will it help in factoring f? So I claim that if you find this g, then you factor f, how? So such a g gives a factor of f, so what

you have is g raised to p – g0 zero mod f. Can you factorize g is to p – g? What is the factorization expression? Just some expression. So claim that this is a factorization.

So if you look at the product g – 0, g – 1 dot dot g – p – 1 right for all the constants if you take the differences and multiply that is the g to the p – g, right. This follows from the Frobenius identity. So x to the p – x. It factorizes into product of x – alpha. So there in x, you substitute g you get this and yeah so that identity does not even need mod f, mod f is just hanging around for free, but now it will be of help.

So what this means is that product of fi's divide product of g – alpha, which means what? So what if you take gcd of g – alpha with f and try this for all alpha, is that any hope of that factorizing f? So when will all those gcd's fail to factorize f? Sorry, no, sure, but we are assuming that f is reducible, otherwise the irreducibility test would have already said irreducible. So we are in the case when f factorizes into at least 2 things, f1 times f2.

So think of the case f1 times f2, k = 2. So f1 times f2 divides this produce g – alpha **"Professor – Student conversation starts."** Either one to divide f1 should divide or f2 should divide Sir? The bad case will be of both of them divide a single g – alpha, right. **"Professor – Student conversation ends."** So basically you try to take gcd's, try gcd of f with g – alpha with respect to the x variable for all alpha.

And if none of these factor f, then it means that in some of the gcd's you are just getting 1 and in some other gcd's you are just getting f, right. You do not get anything between 1 and f. So the time when you get f, it means that f divides that g – alpha. "**Professor – Student conversation starts."** Why is it possible to get f dividing g? Is not the degree of g less than the degree of f? No promises were made. So but we are looking at it mod f anyway, right?

So find all constant g such that g to the power p is equivalent. We are going mod f, yeah so we can assume here find non-constant g and let us add that of degree between 1 and d-1 that we can add. So degree should not be 0 which means that it should not be constant and degree should not be d or more because if it is, then you can reduce mod f and get down to d–1 or less. Yeah okay so with that assumption, yeah that I think then takes care of it.

So then, now you see the simple proof. So clearly since f divides the product all the gcd's

cannot be 1, right, somewhere there should be a non one, but then that non one cannot be f because the gcd will have degree less than d. So something will factor okay. So this implies that there exist an alpha where gcd has degree 1 to d-1. So for some alpha, gcd has to be nontrivial and that factorizes f, right.

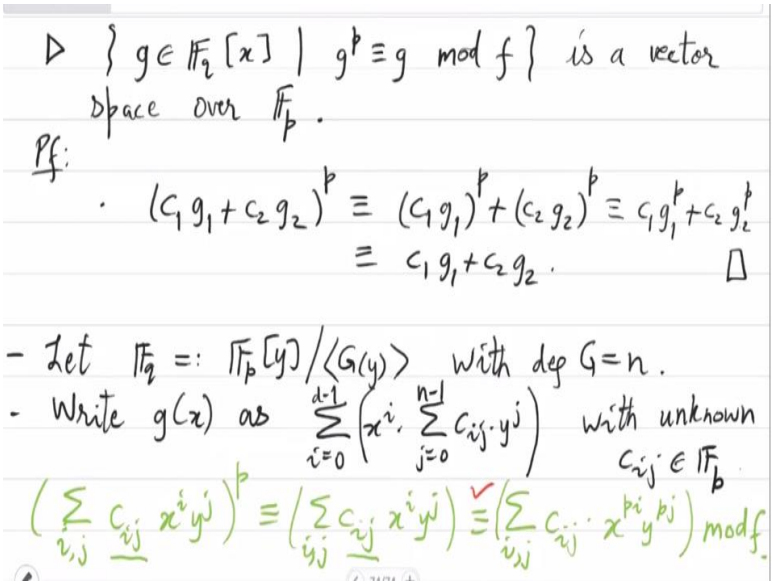
So if you can find this non-constant g limited degree, degree 1 to d-1, then by this process you will be able to factor f, how much time will this process take? How many times are you computing gcd? P times, right, so this process is not the fastest one because it takes at l east p time and p itself may be 1000 bits. So this itself may take 2 raised to 1000 steps, but the same increase is that it is not dependent on q.

So when the characteristic is very small and the field is very large which happens quite often in computer science because p is 2 and the finite field is of size 2 raised to 1000. So in those cases, you can use this idea okay, this is fast. So let us add the time here. So p times gcd and the gcd's degree d so that is around linear and there is a log q okay. So this if you can find g, then on top of that in O tilde p d log q time you can factorize f okay.

"Professor – Student conversation starts." So yes, just one confusion. So these are irreducible factor fi are over fq? Their coefficients are in fq's, fq, cannot g be over fp x such a degree but if it is then. What do you mean g in f px. I mean its coefficient are in fp. Okay, well we have not said anything like that, but fine, so it is not related to this slide. So till now, we have only shown that if you find a g with that green congruence being satisfied, then you will factor efficiently. So g will not be in fq, otherwise how can.

Sorry how can, how can they **((49:46))** module factor. I am not sure f has coefficients in fq. If you take gcd of 2 polynomials, one has fq coefficient, other has fp coefficients. The result will be fp coefficients. **"Professor - Student conversation ends."** Okay, yeah so now we will try to finish this main part which is how do you find this g okay of degree 1 to d-1 satisfying g raised to p = g mod f, any ideas and your allowed time p d log q.

So you have actually enough time to try to solve g raised to p = g for g. How do you do this? So this is a very special system and unless you use that speciality you can never solve it okay.
(Refer Slide Time: 50:56)



So let us see first property which is that the solutions of this, what is the property of the solution space, is it a vector space? Over what? Over which field is this a vector space? Sorry, yes, so it is not Fq, it is Fp. So over this base field, this solution set is actually a vector space because if you take, yeah let us we should do that. So if you look at a solution g1 and a solution g2 and you take combination of this c1 c2 in Fp, what is this raised to p?

So you know that p goes inside and then you know that c1 to the p is you need this to be c1 which is why we said Fp and g1 to the p, g2 to the p you have taken from the solution space, so this c1 g1 + c2 g2 okay. This is the proof. So this solution set is actually a vector space over the base field, prime field, which means that to find all the solutions you just have to compute a basis, right, and so since the solution space is a liner object, this is big development.

So it is not a nonlinear system that you have to solve. There is a lot of linearity in the system. So exactly how you will solve this is again you will use this p exponent, do you want to do this. Yeah, so now let us do everything over fp, so that needs some more notation. So let the finite field F1 be modeled as Fp y mod big G y. So let us pick the specific model of Fq and now the little g it is a polynomial in x of degree d-1 and the coefficients were supposed to be in Fq.

So now, they will be polynomials in y. These are degree of F in all the discussions. So what is the degree of y that is m-1. So let us think of little g as bivariate, i goes from 0 to d-1 that is for x to the i and then there are these coefficients in Fq which again we need to expand as y

polynomials, which goes from 0 to n-1 with unknown cij but they are in Fp okay, is that clear? So little g is the unknown solution we want to find.

And now we are thinking of it or we have actually expressed it as a bivariate individual degree d and n, d-1 and n-1 and the coefficients are unknown, they are numbers, they are 0 to p-1, right. So now we want to find these cij's, so let us do that. What they have to satisfy is this whole thing raised to p has to be the same as itself mod f, right. This is the system. Now in terms of the unknown cij, what do you think is the system, a linear system or a non-linear system.

You remember that non-linear systems are very hard to solve if at all you can solve. Linear systems you can always solve. So what do you think, is this linear or non-linear in terms of the unknows which are cij's? These are the things you want to find. **"Professor – Student conversation starts."** Sorry. Linear, yeah, why? How do you I see it as a linear system. We take any linear combination of constants when raised to the power p will distribute and by this, yeah.

So the p will go inside and it will become cij to the p, but cij to the p is cij, right. **"Professor - student conversation ends."** So the whole systems is actually a linear systems in c's that is it. So the LHS is the same as the last expression cij times x to the pi y to the pj. So now if you look at this congruence, the middle, the second one, this congruence actually gives you a linear system in the unknowns, right.

The dimensions are not very large, so you just solve this. So you solve this linear system, find the basis that is your solution space of g. So you have actually found all the g's and amongst all the g's you just pick something that is non-constant okay. So that is the full Berlekamp algorithm. **"Professor – Student conversation starts."** If we can find the constant, then we are getting something which is congruent to zero mod f so that is becoming our basis right.

Sorry say that again. If we can find out all the cij's, yes, from these equations after solving this equation, yes, so according to this equivalence we are getting a polynomial is congruent to zero mod f right, if the difference. Yes. Well the difference gives you the linear system. **"Professor - Student conversation ends."** Linear system for c bar means that you want to come up with a matrix constants times c bar vector is 0.

If just a matrix times a vector equal to 0 and then you want to find the vector. So you just apply a linear algebra solver or linear system solver on this. How to get the matrix of constant, you just have to compare, I mean you reduce x to the p to the i to degree below d and you reduce y to the p to the j to degree below n and then you compare the monomials both sides. Those are two representations, but they have to be the same.

When they are in the reduced form, they have to the same. So when you compare the monomials, you will get many constraints, but they are all linear constraints, so you saw the linear system. **"Professor – Student conversation starts."** But after computing the constants, how do you get the basis. You know that is the question in linear algebra that you refer to your high school or may be B. Tech courses. **"Professor – Student conversation ends."**

So but we will do the time complexity analysis next time, exactly how will you well or in all these steps overall what is the time complexity it is something like matrix multiplication exponent. Linear system solving is much like multiplying matrices. So you can use fast matric multiplication. We will do that analysis the next time.