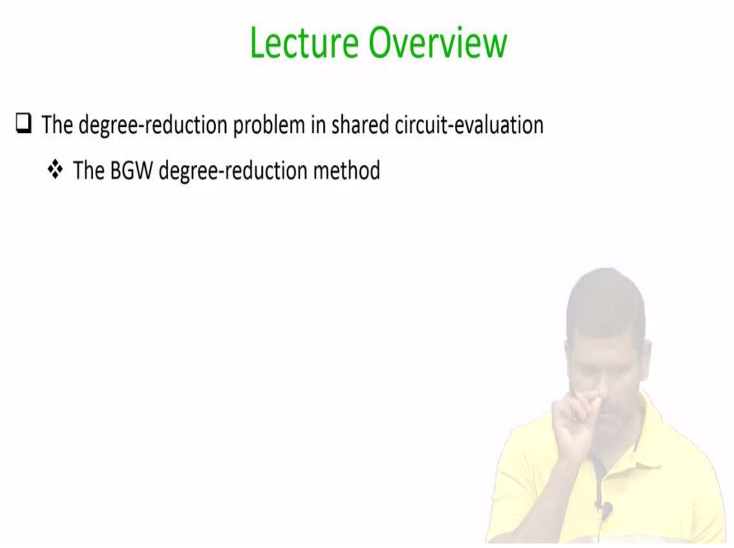
Secure Computation: Part 1

Prof. Ashish Choudhury
Department of Computer Science
Indian Institute of Science – Bengaluru

Lecture – 23
The Degree-Reduction Problem

Hello everyone. Welcome to this lecture.

(Refer Slide Time: 00:33)

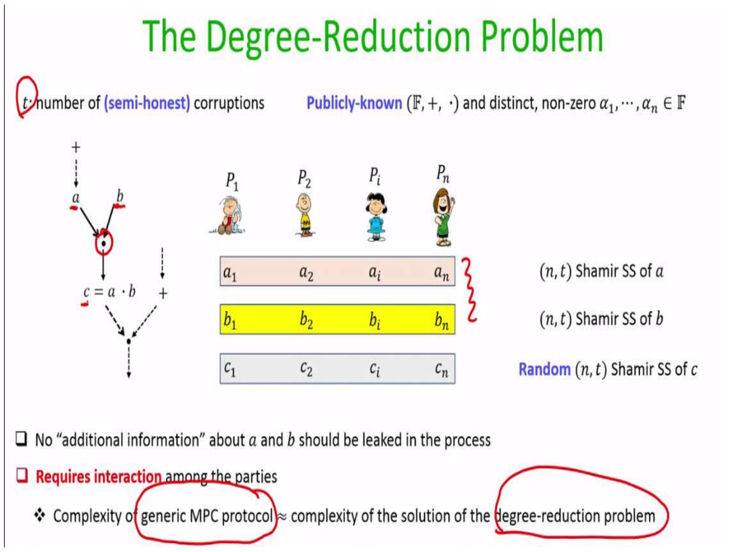


Lecture Overview

- The degree-reduction problem in shared circuit-evaluation
  - The BGW degree-reduction method

So, in this lecture we will introduce the degree reduction problem in the context of shared circuit evaluation where we want to resolve the two problems that we had seen in the last lecture while evaluating a multiplication gate in a secret shared fashion and we will discuss on a very high level BGW degree reduction method.

(Refer Slide Time: 00:56)



The Degree-Reduction Problem

t: number of (semi-honest) corruptions Publicly-known $(\mathbb{F}, +, \cdot)$ and distinct, non-zero $\alpha_1, \cdots, \alpha_n \in \mathbb{F}$

P_1	P_2	P_i	P_n	
a_1	a_2	a_i	a_n	(n, t) Shamir SS of a
b_1	b_2	b_i	b_n	(n, t) Shamir SS of b
c_1	c_2	c_i	c_n	Random (n, t) Shamir SS of c

- No "additional information" about a and b should be leaked in the process
- Requires interaction among the parties
  - Complexity of generic MPC protocol ≈ complexity of the solution of the degree-reduction problem

So, what is the degree reduction problem? So, we are considering a sitting where we have a circuit which the parties are now evaluating as per the BGW approach of shared circuit evaluation in our sitting t is the maximum number of parties which can be corrupted by a semi honest adversary. The value of t will be publically known, but who are the corrupt parties their identity will not be known.

And the circuit is over a finite field our evaluation points are α_1 to α_n or instances of Shamir secret sharing will be executed with respect to these evaluation points and input to the problem are the following. So, you are having a multiplication gate in the circuit and it will be given that the parties hold their respective shares for the inputs of the multiplication gates.

So, the inputs are here the variable say a and b they will be taking some value concrete value from the field based on the concrete value of the parties hold the respective shares of a. So, you can imagine that P_1 has the share a_1 P_2 has the share a_2 P_i has the share a_i and P_n has the share a_n altogether they lie on some polynomial of degree A and suppose they lie on the random polynomial of not degree a degree t.

So, the constant term of the polynomial is a and degree is t it is given that, but no one knows the value of the polynomial as a whole. Only when t + 1 shares are made public the A polynomial can be reconstructed and in the same way another input in the degree reduction problem is the vector of b shares where every party holds the ith component of this vector. So, P_1 holds b_1 P_2 holds P_2 and so on.

So, imagine that these values they lie on the t degree polynomial $B(z)$ which is a random member from the set of all possible polynomials whose constant term is b and degree is t and output that we expect now up at the end of degree reduction is the following. We want that at the end of the degree reduction somehow parties should hold shares c_1, c_2, c_i, c_n when I say parties hold I mean the ith party hold the ith component of this vector.

Such that this vector of values lie on a polynomial C which is a random member of the set of all possible polynomials whose degree is t and constant term is c and that is why the problem degree reduction. Why degree reduction because we had seen in the last lecture that if the parties just locally multiply component wise their respective shares of a and b then the degree of the resultant sharing becomes 2t.

And the resultant vector of shares are not random vector of shares, but we want a mechanism here that the degree of the resultant vector of shares which the parties obtain by locally multiplying their respective shares of a and b are reduced to t and not only that the final vector of shares which the parties have again when I say parties have I mean to say the ith party have the ith component.

But as a whole we demand that the resultant vector of shares lie on a random polynomial of degree t whose constant term is c that means each of the coefficients of this C polynomial except the constant is a random element from the field and in this whole process we do not want any additional information about the a or the input b to be revealed. Otherwise, if I do not put this restriction then it is very easy to do the degree reduction.

A Naïve solution could be for the degree reduction would be the following Naïve solution make public $A(z)$, make public the $B(z)$ polynomial. How we can make them public? We can ask each party to make its shares of a public. We can ask each party to make its respective share of a, b public and then they can use Lagrange interpolation and reconstruct A polynomial and B polynomial.

Once they reconstruct A polynomial and B polynomial then they will learn c because they will be just taking the constant term of A polynomial and the constant term of the B polynomial that will help them to learn c and then they can do the following. Pick a random polynomial of degree t with c as constant term and then compute this vector c 1 to c n that Naïve solution will always work.

But in this process we are actually making the inputs of the gates public which we do not want to do because these inputs a and b are actually the intermediate outcomes of a bigger circuit which we are now evaluating in a secret shared fashion as per the BGW invariant and remember in the BGW invariant we only want the final outcome to be made public not the intermediate outcomes.

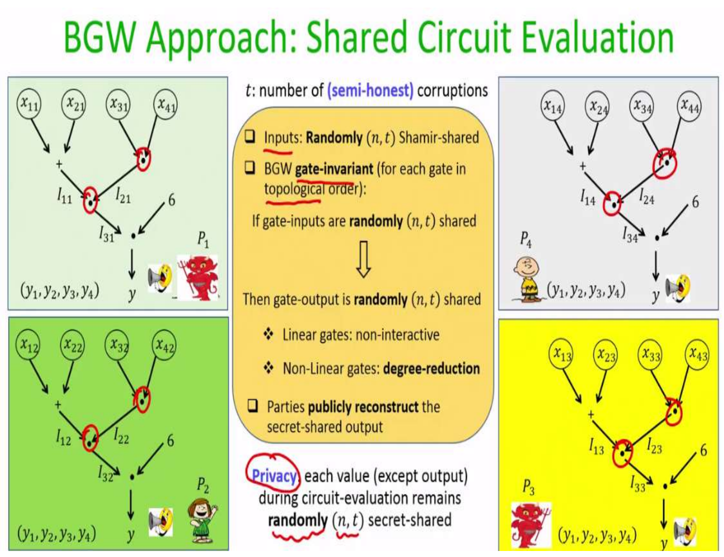
So, we cannot afford to make this values a and b public and hence this Naïve solution will not work, it would not maintain the overall privacy of the computation. So, the challenge is to reduce the degree of the sharing and in the process we do not want to leak anything about the inputs a and b and this requires interaction among the parties this process of taking this vector of a shares, vector of b shares and then somehow obtaining this random vector of c shares it requires interaction among the parties.

And that is why when it comes to the complexity of any generic MPC protocol when I say complexity by that I mean the number of times the parties have to interact and how much they have to communicate that is what I mean by the complexity namely the round complexity and communication complexity. So, the communication complexity and round complexity of any generic MPC protocol is actually proportional to the round complexity.

And communication complexity of the degree reduction problem because wherever there will be a multiplication gate in the circuit which parties want to securely evaluate. They have to deploy this degree reduction method to maintain the BGW gate invariant that means if we want to ensure that starting with the shares of a and starting with the shares of b whose degree of sharing is t somehow the parties should obtain shares of c lying on a t degree polynomial and that too on a random t degree polynomial we have to solve an instance of degree reduction problem.

And wherever there is an occurrence of the multiplication gate we have to deploy a solution of the degree reduction problem.

(Refer Slide Time: 09:18)


BGW Approach: Shared Circuit Evaluation

t: number of (semi-honest) corruptions

- ❑ Inputs: **Randomly** (n, t) Shamir-shared
- ❑ BGW **gate-invariant** (for each gate in topological order):

  If gate-inputs are **randomly** (n, t) shared

  ⇩

  Then gate-output is **randomly** (n, t) shared

  - ❖ Linear gates: non-interactive
  - ❖ Non-Linear gates: **degree-reduction**
- ❑ Parties **publicly reconstruct** the secret-shared output

Privacy: each value (except output) during circuit-evaluation remains **randomly** (n, t) secret-shared

x_{11} x_{21} x_{31} x_{41}
I_{11} I_{21} I_{31} 6
P_1
(y_1, y_2, y_3, y_4) y
x_{14} x_{24} x_{34} x_{44}
I_{14} I_{24} I_{34} 6
P_4
(y_1, y_2, y_3, y_4) y
x_{12} x_{22} x_{32} x_{42}
I_{12} I_{22} I_{32} 6
P_2
(y_1, y_2, y_3, y_4) y
x_{13} x_{23} x_{33} x_{43}
I_{13} I_{23} I_{33} 6
P_3
(y_1, y_2, y_3, y_4) y

So, what I am saying is if I take this concrete circuit I have two multiplication gates here in the circuit. By the way this gate even though this is a multiplication gate it is a multiplication by a public constant 6 or a public constant c where c is a field element hence it will be considered as a linear gate it is not a multiplication gate. So, to maintain the BGW gate invariant we have to start with secret sharing the inputs by asking the respective parties to share them by instance by running instances of Shamir secret sharing.

So, every party will obtain their respective shares of the input this will be done for all the input gates and then the parties will start maintaining the BGW gate invariant where they will start scanning the circuit layer by layer by layer. So, we often say scan the circuit in topological order and somehow we have to maintain the invariant that if the inputs of the gate that we are considering right now when I say v I mean collectively the parties are considering,

And that is why it is important that the protocol is executed in the synchronous environment. So, if the parties are considering a particular gate. So, to start with they will start with this plus gate because they will be first waiting for the input being secret shared. Once all the inputs are secret shared then they will come to the next layer. In the next layer the current gate is this plus gate so they will check whether they have the shares for this plus gate.

Right now all the parties have their respective shares of plus gate they will just add them respectively, locally and then they will obtain their respective shares of this intermediate value then they will shift their focus to the next gate which is a multiplication gate. For this multiplication gate the parties have their respective shares of x 3 and x 4 and now they cannot just locally multiply their respective shares of x 3 and x 4 and proceed to evaluate the next gate because we had seen that there are two problems associated.

The degree of sharing will blow up now it become 2t and the resultant shares of I 2 which the parties will now have would not be lying on a random 2t degree polynomial. So that means to evaluate this multiplication gate the parties have to jointly run an instances of degree reduction problem they have to run an instance of solving the degree reduction problem.

And after solving the degree reduction problem they will finally have their respective shares namely I 21, I 22, I 23 and I 24 which collectively constitute a vector of n, t random shares for the value I 2. Now once this multiplication gate has been evaluated that means this layer has been taken care by all the parties now they will focus to the next layer. In the next layer again they have a multiplication gate.

And now they will consider this instance as a fresh instance of the degree reduction problem that means now they have their respective shares of I 1 and I 2 where I 1 and I 2 is the intermediate value, but right now each party has their respective shares of I 1 and each party has its respective share of I 2 and now collectively the shares of I 1 and I 2 are random shares whose degree is lying on a t degree polynomial and constant term being I 1 and I 2.

And now they run an instance of solving the degree reduction problem and obtain their respective shares of this intermediate outcome I 3 that means now what I am saying is starting with I 11, I 12, I 13 and I 14 and I 21, I 22, I 23, I 24 they solve an instance of degree reduction problem and obtain this shares I 31, I 32, I 33 and I 34 respectively and in this process nothing about the actual value of I 1 and nothing about the actual value of I 2 is revealed.

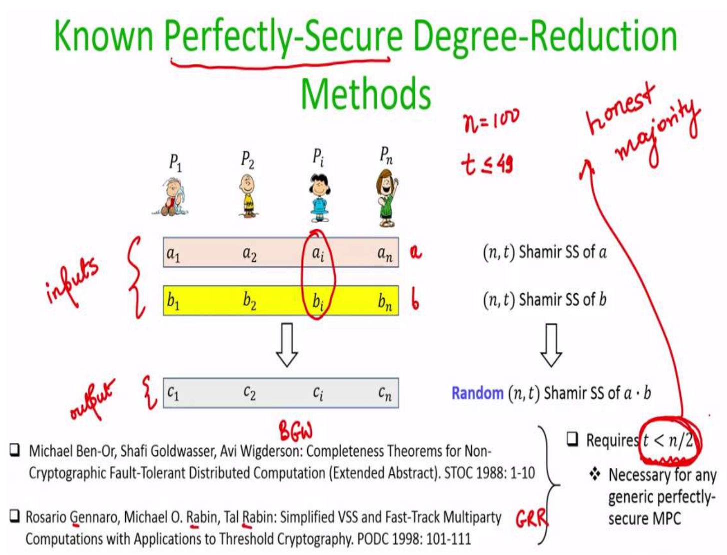
And this vector of four shares which now the parties obtain after solving the degree reduction problem lie on a t degree polynomial which is a random polynomial whose constant term is I 3. Now once this gate is evaluated they go to the next layer the next layer is actually consisting of a linear gate namely multiplying it with a public constant and they will be knowing the value of the public constant in this case it is 6.

So, every party locally multiplies its respective share of this intermediate outcome because that is the gate input with this constant 6 and they will obtain their respective shares of the y value and that will ensure that all the gates are evaluated so as we had seen in this example the linear gates namely this gate and this gate for these two gates the gate invariant can be maintained locally it will not require any interaction.

But for the non-linear gates namely this gate and this gate the parties have to collectively solve two instances of degree reduction problem and then finally the parties after evaluating all the gates will publically reconstruct the secret shared output and now we can argue about the privacy, we can argue, we can say that entire process maintains the privacy property because each value in this entire process of shared circuit evaluation remains n, t secret shared.

And not only n, t secret shared they are randomly n, t secret share namely at each instance at each layer not at each instant at each layer we have ensured that the resultant vector of values, the resultant vectors of share together lie on a random polynomial of degree t whose constant term would have been the actual value of that actual intermediate value and this whole thing works under the assumption that we have a method of solving the degree reduction problem.

(Refer Slide Time: 16:56)



Known Perfectly-Secure Degree-Reduction Methods

$n=100$
$t \leq 49$
honest majority
P_1 P_2 P_i P_n
inputs
a_1 a_2 a_i a_n a
b_1 b_2 b_i b_n b
output
c_1 c_2 c_i c_n
(n, t) Shamir SS of a
(n, t) Shamir SS of b
Random (n, t) Shamir SS of $a \cdot b$
BGW
GRR

- Michael Ben-Or, Shafi Goldwasser, Avi Wigderson: Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation (Extended Abstract). STOC 1988: 1-10
- Rosario Gennaro, Michael O. Rabin, Tal Rabin: Simplified VSS and Fast-Track Multiparty Computations with Applications to Threshold Cryptography. PODC 1998: 101-111

- Requires $t < n/2$
  - Necessary for any generic perfectly-secure MPC

So, now we will see the known perfectly secure methods of degree reduction problem. Why perfectly secured? Because we are now considering a scenario where our adversary is computationally unbounded and this is one instance of the degree reduction problem. So, these two vectors are the inputs that means parties collectively hold these two vectors with ith party holding the ith component of these two vectors.

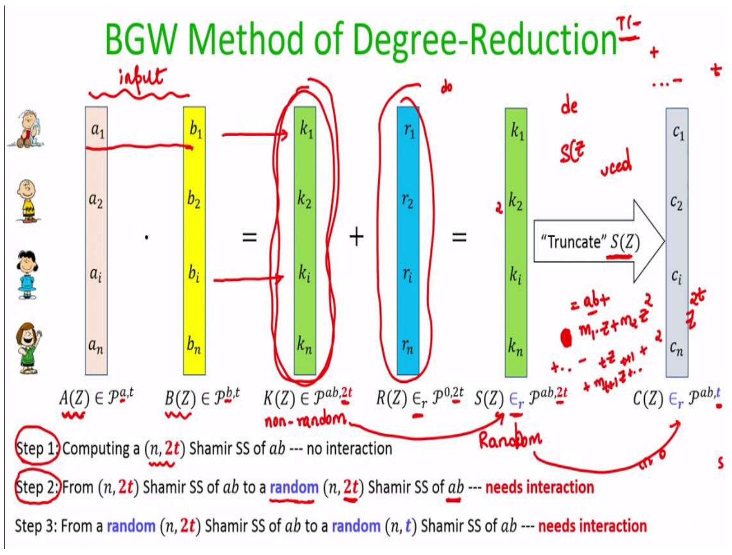
And now we want output of vector of random shares lying on a t degree polynomial and whose constant term would have been the product of inputs here. So, there are two very well known methods for solving this degree reduction problem. The original BGW protocol and their paper they proposed a solution for this degree reduction problem and later that method was simplified in this second paper.

This second method is often called as the GRR degree reduction method attributed to Gennaro, Rabin and Rabin. Both these methods require a number of corrupt parties to be strictly less than n over 2. So, this term is often called as this condition $t < n / 2$ is often called as honest majority. What do I mean by honest majority? N is the total number of parties we require that the number of bad parties who are collectively trying to infer something should be strictly less than $n / 2$.

So, if n is equal to say 100 t can be maximum 49. So, that is why majority of the parties will be the good parties they would not be under the control of the adversary that is why this setting is called as the honest majority setting. Now, you might be wondering that why this restriction? Is this condition necessary only with respect to these two methods of degree reduction?

The answer turns out to be no. It turns out that if you take any generic perfectly secure MPC protocol and if you want to securely compute multiplication then we necessary require this honest majority condition. So, this honest majority condition is not enforced just because of the BGW or the GRR method of solving the degree reduction problem. It is attributed to it is an inherent requirement for any generic perfectly secure MPC protocol and we will prove the necessity of this bound later.

(Refer Slide Time: 20:08)



BGW Method of Degree-Reduction

input
a_1 a_2 a_i a_n
b_1 b_2 b_i b_n
k_1 k_2 k_i k_n
r_1 r_2 r_i r_n
c_1 c_2 c_i c_n
"Truncate" $S(Z)$
$A(Z) \in \mathcal{P}^{a,t}$
$B(Z) \in \mathcal{P}^{b,t}$
$K(Z) \in \mathcal{P}^{ab,2t}$
non-random
$R(Z) \in_r \mathcal{P}^{0,2t}$
$S(Z) \in_r \mathcal{P}^{ab,2t}$
Random
$C(Z) \in_r \mathcal{P}^{ab,t}$
$= ab + m_1 \cdot Z + m_2 Z^2 + \dots + m_{t+1} Z^{t+1} + \dots$

Step 1: Computing a $(n, 2t)$ Shamir SS of ab --- no interaction

Step 2: From $(n, 2t)$ Shamir SS of ab to a random $(n, 2t)$ Shamir SS of ab --- needs interaction

Step 3: From a random $(n, 2t)$ Shamir SS of ab to a random (n, t) Shamir SS of ab --- needs interaction

So, I would not be going into the full fledge details of the BGW reduction method, but I will be giving you the high level over that what exactly happens in the method. So, to begin with this is your input the parties hold their respective shares of some a values and respective shares of b value. The A values lie on some A polynomial A of Z polynomial the B values for the B shares lie on a polynomials B of Z.

Both of them are t degree polynomial it is known to everyone and a might not be known and b might not be known. Now to obtain finally a vector of shares lying on a random polynomial of degree t with constant tem being c the parties do the following. The first step is they non interactively compute a set of n, 2t secret sharing of a, b. How can they do that? They have to just locally multiply their respective shares of a and b.

So, P 1 it takes a 1 b 1 multiply and it obtains k 1 P 2 takes a 2 b 2 multiplies and obtains k 2 so that is why it is a local operation. It does not require interaction because P 2 owns both a 2 and b 2 P i it owns both a 1 and b i. It multiplies and obtain a value called k 1. So, now if I consider this vector of n values k 1, k 2, k i, k n. They together lie on a polynomial say the K polynomial whose degree is 2t and whose constant term is ab.

By the way in this entire process I am fixing the evaluation points to be alpha 1, alpha 2, alpha i, alpha n that means when I say the A polynomial evaluated at alpha 1, alpha 2, alpha i, alpha n gives you this vector of a shares, the B polynomial evaluated at alpha 1, alpha 2, alpha i, alpha n gives you this vector of b shares and so on. Now the first step in the BGW reduction method is to take this vector of shares lying on a 2t degree polynomial whose constant term is ab.

And convert it into another vector of shares lying on a random 2t degree polynomial and whose constant term is still ab. So, right now this K polynomial is not a random polynomial because say for instance it is a reducible polynomials it is not a random member from the set of all possible polynomials of degree 2t with constant term being ab. So, as part of the degree reduction the first step that the parties are doing is they are doing the randomization here.

What do I mean by randomization? They somehow convert this vector of values k 1 to k n into another vector of values k 1 to k n into another vector of values r 1 to r n which now lie on a polynomial say R polynomial whose degree is 2t sorry they do not convert this into vector of (()) (24:10). So, eventually the goal is to convert this vector into another vector of shares for the value ab.

But that resultant vector of shares should be a random vector of shares so how do you they do that? So, the parties first generate a vector of shares for the values 0 lying on a 2t degree polynomial and is vector of shares should be random that means this R polynomial should be random that means each coefficient of this R polynomial should be a random element from the field, it could be any random coefficient except the constant term which is 0.

So, how exactly this R polynomial will be generated and in the process as a vector no one will know the full vector parties will just have the respective components. So, whenever I am denoting a vector in a box here that means as a entire vector it is not known publically parties only know the respective components of the vector. So, the first step will be the parties generate this random vector of shares for the value 0 lying on a 2t degree polynomial.

And this requires interaction it is not that magically they obtain this random vector of shares parties interact among themselves. Each party pick some value and exchange messages and finally together they hold this vector and now if the parties have this vector if the component wise go and add their respective shares of K polynomial K polynomial with their respective shares of R polynomial what they will obtain?

They will obtain a vector of shares. So, sorry for the notational overhead I should have called it some new vector of values because k I have already used. So, let us call them as S 1, S 2, S i, S n. So, they take their respective shares from this K polynomial and their respective shares of the R polynomial locally add them, they will now obtain collectively a vector of n values whose degree will be 2t.

Why degree will be 2t? Because K polynomial was a 2t degree polynomial, R polynomial was a 2t degree polynomial and if they are collectively and locally not collectively they are locally adding these component why is these two vectors the resultant vector of values will also constitute distinct points on a 2t degree polynomial and what will be the constant term of that polynomial that will be ab because this S polynomial is summation of the K polynomial and R polynomial the constant term of the K polynomial was ab.

The constant term of the R polynomial was 0 so that is why the constant term of the S polynomial will be ab + 0 and ab + 0 will be ab, but now the important thing is that now this is a random polynomial. Why this is a random polynomial? Because even though the K polynomial was not random that means the coefficients of the K polynomial not random element from the field.

The coefficients of the R polynomial are random elements from the field and if you component wise add the coefficients of the K polynomial and the R polynomial that will give you the respective coefficients of the S polynomial since the random coefficient is added to a non random coefficient the overall coefficient will be a random coefficient and if the S polynomial is a random polynomial then I can say that this resultant vector of shares constitute a random n, 2t sharing of the value ab.

So, what we have done in step 1 basically is we have taken a vector of shares for ab lying on a non random 2t degree polynomial and converted into another vector of n shares of ab lying on a random 2t degree polynomial by adding this random vector of shares component wise for the value 0, but our degree is still 2t. So, the degree if finally reduced in the second step of the degree reduction.

So, step 1 actually does not require any investment the actual degree reduction starts from step 2 here which I call step 1 of the degree reduction. So, in the step 1 or in the stage 1 of the degree reduction the parties first randomize the underlying secret sharing and in stage 2 they do the degree reduction that means they convert this vector of shares S 1, S 2, S i, S n into another vector of shares C 1, C 2, C i and C n which now lie on a t degree polynomial with constant term being ab.

But now this C polynomial is a random member from the set of all possible polynomials of degree t with ab being the constant term that means each and every coefficient of this C polynomial are random except the constant coefficient and this step again requires interaction among the parties and basically in this degree reduction this stage 2 degree of S Z is reduced. In stage 2 basically the idea is to truncate the polynomial S Z what do I mean by truncating?

So, remember this S polynomial is a 2t degree polynomial. So, it is of the form it is constant term is ab and you have the remaining coefficient so you have S 1 let us not call it S 1. Let us call it say call m 1 times z + m 2 times z square + m t times z t + say you have the remaining coefficient m t + 1 z power t + 1 and I have 2tth coefficient for the z power 2t that is the form of S Z polynomial. It has total 2t + 1 coefficients.

The constant coefficient is fixed to be ab remaining all other coefficients are random elements from the field that is how that will be the structure of this S polynomial and no one will know the S polynomial and no one will know the S polynomial in clear parties just hold the shares of S polynomial. What can I say about the truncated polynomial T z consisting of only the first t + 1 coefficient of this S polynomial.

That means I am focusing only the first t + 1 coefficient starting with the power of Z power 0. The claim is since this S polynomial was a random 2t degree polynomial whose term was ab. I can claim that this t polynomial is a random t degree polynomial whose constant term is ab because I am not changing the coefficient m 1, m 2, m t I am taking them as it is and I am making the remaining coefficients the higher order coefficient of S polynomial to be 0 value that is what I mean by the truncate S Z.

So, in this stage of the degree reduction the idea is to somehow take this vector of shares. No one knows the S polynomial remember and no one learns the T polynomial as well. All this computation that means truncating the S polynomial into the T polynomial is actually happening by performing operations on the shares that means somehow we have to ensure that without knowing the S polynomial.

But just having each party owning the respective shares of the S polynomial somehow the parties should obtain another vector of values which should lie on this T polynomial where the T polynomial is actually a truncated version of this S polynomial and this can be done very easily basically the parties each party has to secret share its S share here of the S polynomial.

So P 1 will secret share S 1, P 2 will secret share S 2, P i will secret share S i and P n will secret share S n and it is easy to see that the mapping from this S polynomial t the T polynomial namely the truncate operation is a linear function namely I can express the coefficients of the T polynomial as a linear function of the coefficients of the S polynomial that mapping is a linear function.

And we know how to compute linear functions securely. So, to compute that function securely basically it is fine if we know the coefficients of S or equivalently if we know the distinct points on the S polynomial both are fine. Since the coefficients of the S polynomial are not known to anyone, but the points on the S polynomial are available with the respective parties if the party secret share those points namely S of alpha 1, S of alpha 2, S of alpha n.

My claim is the t of alpha 1, t of alpha 2, t of alpha n is a linear function of S of alpha 1, S of alpha 2, S of alpha n and basically in this stage 2 we are computing that linear function. So, that will again require interaction and once we do this truncate operation parties will now have their resultant shares lying on a t degree random polynomial with constant term being ab.

So, now you can see that what are the two stages here involved in this degree reduction method. So, let me summarize what we have discussed in this slide. The parties first obtained shares lying on a non random 2t degree polynomial, but constant term being a times P. They first convert it into a random polynomial of degree 2t the degree does not change. Here also it was 2t here also it is 2t.

It is just that here the shares lie on a non random polynomial of degree 2t somehow magically they make it they convert those shares into another set of shares which now lie on a random polynomial of degree 2t how they do that? By respectively adding shares of 0 where it will be ensured that the shares of 0 lie on a random polynomial of degree 2t that is step 1 doing the randomization and then in stage 2 we do the degree reduction.

So, with that I end this lecture. I have not discussed the full fledged details of the BGW reduction, but I am sure that the high level discussion that we had is enough or sufficient for you to go back and read the BGW paper and understand the degree reduction solution that they have proposed. Thank you.