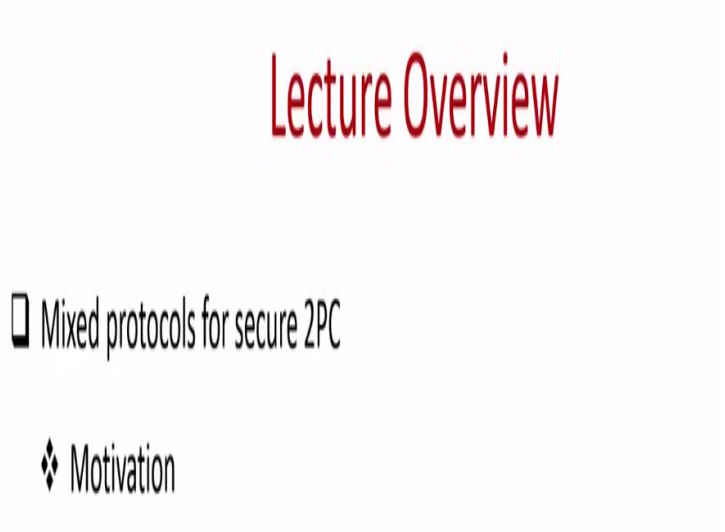
Secure Computation – Part 1
Prof. Ashish Choudhury
Indian Institute of Technology – Bangalore

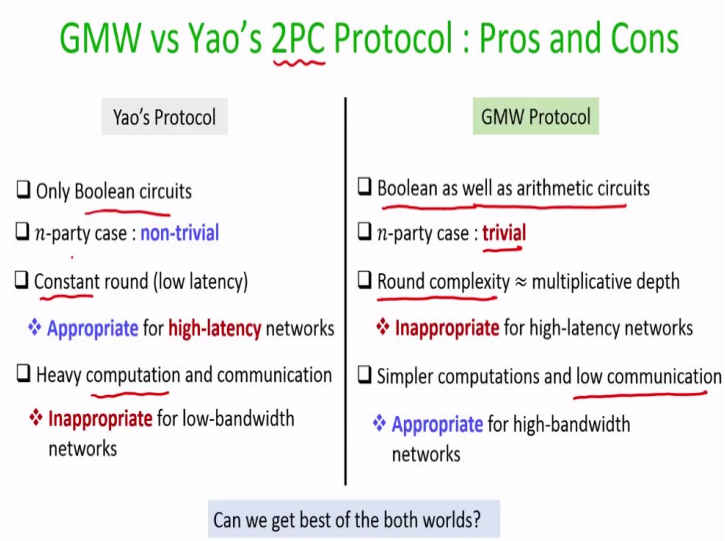
Lecture – 53
Mixed Protocols for Secure 2PC

(Refer Slide Time: 00:32)



Hello everyone, welcome to this lecture. So, in this lecture we will start discussing about mixed protocols for secure two-party setting and this is very practically motivated. So, we will first see the actual motivation for this, right.

(Refer Slide Time: 00:45)



So if we consider two-party computation protocols, then we have two different paradigms. We have Yao's protocol and we have GMW protocol, each of them has its own pros and

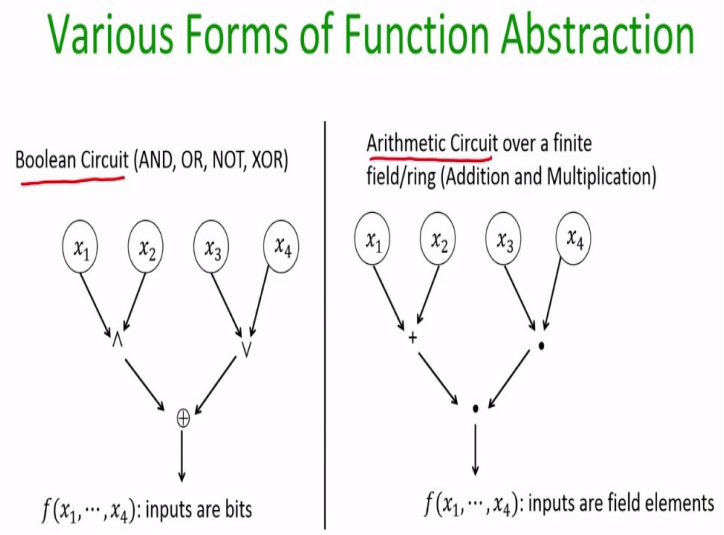
cons. Yao's protocol is tailor made only for Boolean circuits, whereas GMW protocol can be used both for evaluating Boolean circuits as well as arithmetic circuits. Yao's protocol is tailor made for the two-party case, the n-party case is non-trivial.

Whereas the GMW protocol can be easily extended to the n-party case. Yao's protocol always requires a constant number of rounds of interaction and hence it is appropriate for high-latency networks namely where the delay round trip delay is extremely large and that is why the parties cannot afford to interact very often. Whereas GMW protocol can be used where round is not a concern or interactions is not a concern.

Because the round complexity of GMW protocol is proportional to the multiplicative depth and hence if you are using a high-latency network then it is inappropriate to use the GMW protocol. The Yao's protocol requires heavy computation and communication and the amount of communication is enormously large because for every gate in the circuit a garbled gate has to be communicated by the garbler which consists of 3 or 4 ciphertext depending upon the type of gates and that requires enormous amount of bandwidth.

Whereas GMW protocol requires performing very simple computations during circuit evaluation and it requires very low communication and that is why it is highly appropriate for high bandwidth network. So now the question is can we get best of both worlds? And that is precisely the motivation of Yao's mixed world computation.

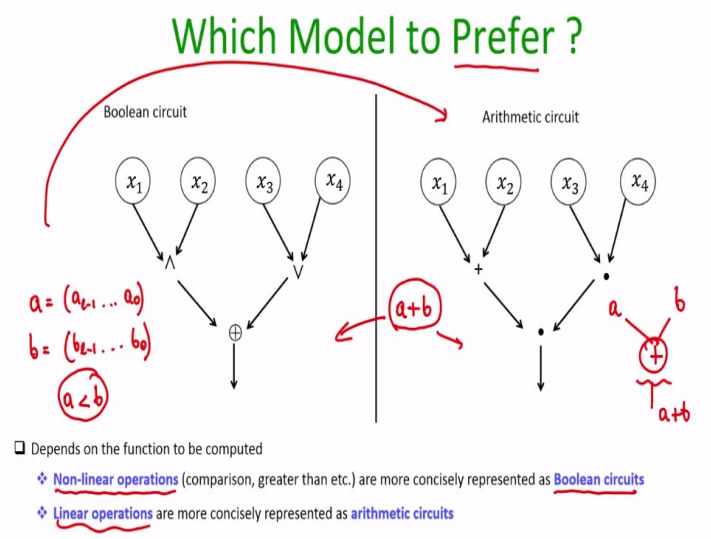
(Refer Slide Time: 03:02)



So, to understand the mixed world computation in a better way let us go back to the first lecture, where we studied the various dimensions in which we can study the MPC problem. And there we discussed that there are two forms of function abstraction which are commonly used in the literature. The first one is the Boolean circuit abstraction where we assume that the circuit or the function which parties want to evaluate is represented by a publicly known Boolean circuit.

Whereas the other abstraction is the arithmetic circuit abstraction where we assume that a function which the parties want to securely compute is represented as an arithmetic circuit over a finite field or a ring or some appropriate algebraic structure, where we can perform both addition as well as multiplication.

(Refer Slide Time: 04:03)



So, now the question is which model to prefer? Whether it is always preferable to follow the Boolean circuit abstraction or whether it is always preferable to follow the arithmetic circuit abstraction? And the; answer to this question namely which model to prefer depends upon the function to be computed. What does that mean? If there are nonlinear operations which are involved in your function or computation.

By nonlinear operation I mean say comparing two values, two numbers, finding out which number is greater and so on. These kinds of operations are the nonlinear operations and they are more concisely represented as Boolean circuits. So if I have a value a which is an l-bit number and value b which is another l-bit number and I want to check whether a is less than b securely.

Then for that, I have to assume that there is a circuit representing the computation a less than b or not. So, if I try to represent this computation a less than b as a Boolean circuit, then it can be represented very concisely, it will have only a very small number of gates. I can represent the same computation a less than b even as an arithmetic circuit as well where all the computations are actually the plus and multiplication operations of the underlying field.

But if I do that, then the number of gates which will be there in the resulting circuit will be large compared to the Boolean circuit representation of a less than b. So, now if I have a concise representation in the Boolean world for this function a less than b, then I can use the corresponding Boolean world MPC protocol. If I want to use Yao, I can use Yao. If I want to use GMW, I can use GMW.

But if I represent it as an arithmetic circuit, then the circuit will be enormously large and then I have to use only the GMW protocol because I cannot run the Yao's protocol. Whereas, if I have operations which are linear operations, say for instance adding the integers a and b then again I can represent computation a + b both by a Boolean circuit as well as an arithmetic circuit.

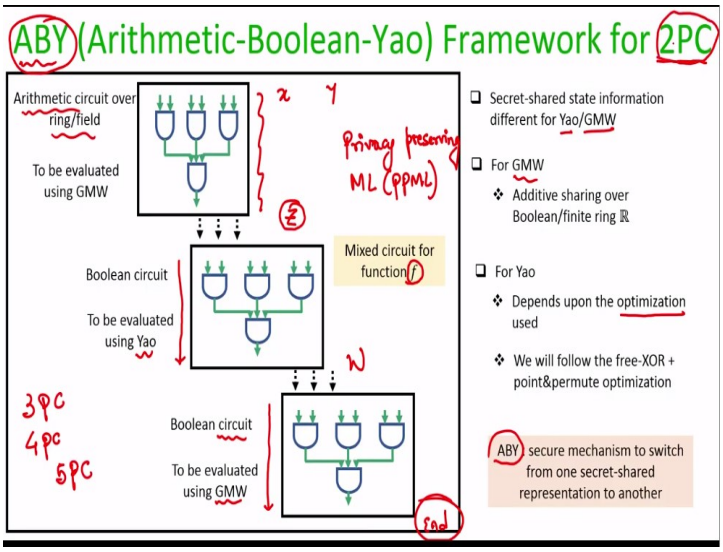
But in the Boolean circuit representation, the number of gates which are used to represent this computation a + b will be large compared to the number of gates which I require to represent a same computation in arithmetic world. More precisely in the Boolean circuit representation, I will be considering the bit by bit addition and I have to take care of the carry and overflow and so on.

So, there will be one bit adder composed l number of times with a logic to take care of the carry and so on and that circuit needs to be evaluated using a MPC protocol for evaluating Boolean circuit if I want to compute this function in the Boolean world. But if I represent his circuit or this computation in the arithmetic world and this is just a simple plus operation over an appropriate ring or a field, where the inputs are say l-bit numbers a and b and output is a + b.

And hence, you can see now that my circuit is so compact here compared to the Boolean circuit representation. So, it depends upon what kind of computations are involved which

actually dictates which model to prefer. You cannot always say that the Boolean circuit abstraction is better compared to the arithmetic circuit abstraction.

(Refer Slide Time: 08:10)



So, motivated by this, we will now discuss what we call as the ABY framework for secure 2PC. ABY stands for Arithmetic Boolean and Yao. And the idea here is that instead of representing the function which the two parties want to securely compute always as a Boolean circuit or as an arithmetic circuit, we identify which part of the computation can be concisely represented in terms of either a Boolean circuit or as an arithmetic circuit.

So, we will assume that we no longer have a uniform circuit. By uniform circuit I mean a circuit which is completely a Boolean circuit or a circuit which is completely an arithmetic circuit. It is now a mixed circuit depending upon what part of the computation can be compactly or concisely represented as an arithmetic circuit or a Boolean circuit. So, for instance, you can imagine that there is an example function.

Example computation where say the parties have identified that the first few parts of the, first few steps of the computation can be compactly represented as an arithmetic circuit over some ring or a field and they would like to evaluate that part of the computation using the GMW protocol. Now without disclosing the result of the intermediate outcome because that is not the end of the computation.

The computation is ending here without disclosing the intermediate results, they will now like to switch to Boolean circuit representation and would like to compute the next few steps of

the computation as per se the Yao's protocol. And then again say the last part of the computation can be evaluated or represented as a Boolean circuit and they are now interested to evaluate the last part of the circuit or the computation or perform the last part of the computation using the GMW protocol.

So that is what we are now going to look for aiming. We are now going to aim for this mixed world computation, mixed circuit representation for the function. But the challenge here is that the secret sharing state information or secret sharing semantics for the Yao's protocol and for the GMW protocol are different. More specifically, for the GMW world, the values are as per the additive secret sharing, over the ring.

Whereas for the Yao secret sharing protocol depending upon what kind of optimization you are using, the interpretation is completely different. It is no longer the case that the share of the two parties if you sum them you get the actual value, which is the case for the GMW secret sharing semantics. So what exactly is this ABY framework does, it provides you a secure mechanism to switch from one secret sharing representation to another representation.

Say for instance, if we start the computation with inputs x and y here and say the intermediate computation here is z, where z is a vector of bits you can imagine, then what we want to do here is without disclosing the value of z, we would like to switch over from the GMW secret sharing representation of z to the Yao secret sharing representation of z and then evaluate this part of the computation as if the values in z are Yao secret share.

And now say w is the result of this intermediate computation, where the values in w the vector w is Yao secret shared, what now we want to do is without disclosing the actual values in the vector w, we would like to switch over from the Yao secret sharing representation of w to GMW representation of the contents of w and compute this remaining part of the circuit in the GMW domain and then finally reconstruct the values as for the GMW reconstruction protocol.

That is what your ABY framework allows. And this is very practically motivated specifically in the context of privacy preserving machine learning, PPML because typically ML algorithms they involve different types of computation, they have both linear computations as

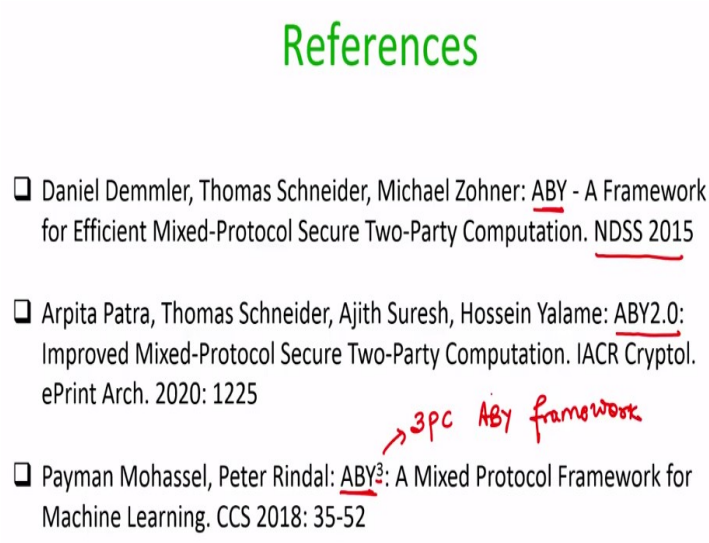
well as nonlinear computations involved. And the idea will be that whenever you want to do machine learning computation in a privacy preserving way, we have two options.

Either we can perform the computation as per the Yao's protocol or we can perform the computation as per the GMW protocol, but that will give us different trade-offs. What we can instead do is if we want to do in a more efficient way, then what we can do is we can identify part of the ML computations which can be represented compactly as per the Boolean circuit abstraction, the part of the computation which can be abstracted as per the arithmetic circuit abstraction.

And depending upon what is my circuit abstraction I am following in the entire ML computation, I can use this ABY framework and instead of evaluating the entire circuit as per only Yao or as per only GMW, we can switch from one representation to another representation and perform the ML computations in a more efficient way. And that is why this ABY framework is very practically motivated.

We are going to discuss this framework in the context of two-party computation, but this framework is now a standard event to the three-party case, four-party case and some of the works are even trying to extend these techniques to the five-party case. But due to interest of time, we will be just focusing only on the two-party scenario.

(Refer Slide Time: 14:28)



So, these are the references which I used for today's lecture. This ABY framework for the two-party case study was initiated in this work. So that was ABY version number 1.0. And

now we have more efficient versions of ABY switching mechanisms due to a very recent work. And as I said that we also have ABY conversions in the three-party case. So, this superscript 3 denotes the 3PC ABY framework. And after this work there are more efficient works which have come out which propose more efficient 3PC ABY conversion system. Thank you.