CS 65500
Advanced Cryptography

Lecture 4: Semi-Honest GMW - I

Instructor: Aarushi Goel

Spring 2025

Agenda

- Secret Sharing Schemes
- Secure Two-Party computation of linear functions
- GMW Protocol

Reminder: HW1 is due tonight!

Semi-Honest Secure Two-Party Computation

Definition: A protocol π securely computes a function f in the semi-honest model, if $\exists$ a pair of two n.u. PPT simulator algorithms S_A and S_B, such that for every security parameter K, and $\forall$ inputs $x, y \in \{0,1\}^K$, it holds that:

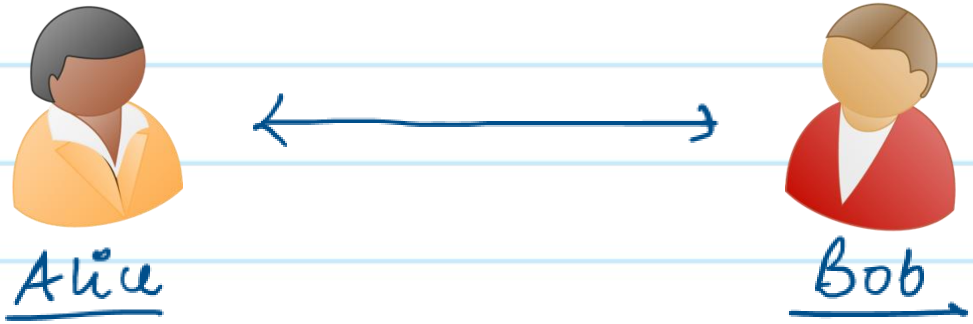
$$\{S_A(x, f(x,y)), f(x,y)\} \approx \{\text{view}_A(e), \text{out}_B(e)\}$$

$$\{S_B(y, f(x,y)), f(x,y)\} \approx \{\text{view}_B(e), \text{out}_A(e)\}$$

where $e \leftarrow [A(x) \leftrightarrow B(y)]\}$

Oblivious Transfer (OT)

Also called 1-out-of-2 OT:



Input :	(a_0, a_1)	b
Output :	$\perp$	a_b

Security: Alice doesn't learn b.

Bob doesn't learn a_{1-b}

Oblivious Transfer (OT)

Can be generalized to 1-out-of-K OT:

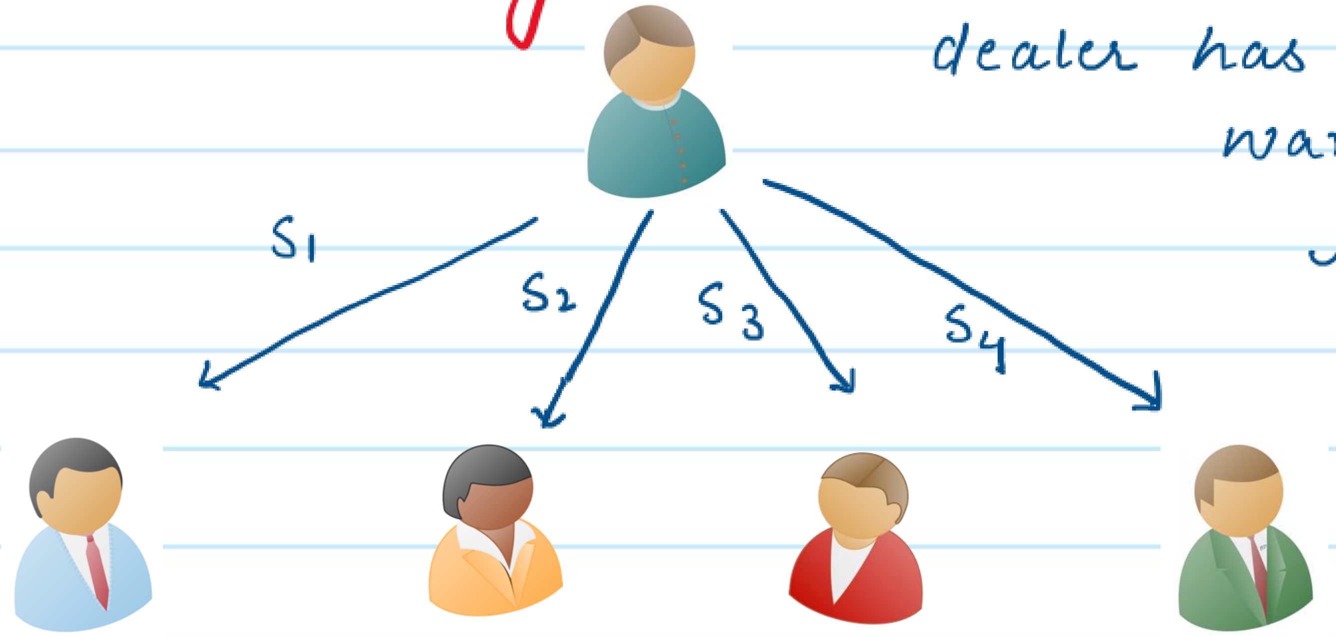
Input :	$(a_1, a_2, \ldots a_K)$	$b \in [K]$
Output :	$\perp$	a_b

Security: Alice doesn't learn b.

Bob doesn't learn $\{a_i\}_{i \neq b}$

Exercise: Think about how you can use 1-out-of-2 OT to design 1-out-of-K OT.

Secret Sharing (k, n)



dealer has a secret s and wants to divide it into n parts/shares

Correctness: Any subset of k shares can be combined to reconstruct the secret s.

Security: Any subset of $\leq k-1$ shares reveal no information about the secret s.

Secret Sharing (k, n)

Definition: A (k, n) secret sharing consists of a pair of PPT algorithms (Share, Reconstruct) s.t.,

- Share$(s) \longrightarrow (s_1, \ldots, s_n)$

- Reconstruct$(s'_{i_1}, \ldots, s'_{i_k})$ is such that, if $\{s'_{i_1}, \ldots, s'_{i_k}\} \subseteq \{s_1, \ldots s_n\}$, then it outputs s.

- $\forall s, s'$ and for any subset of at most $k-1$ indices $X \subset [1, n]$, $|X| < k$ the following distributions are statistically close:

$$\{(s_i \mid i \in X) \; ; \; (s_1, \ldots, s_n) \leftarrow \text{Share}(s)\},$$
$$\{(s'_i / i \in X) \; ; \; (s'_1, \ldots, s'_n) \leftarrow \text{Share}(s')\}$$

(n, n) Secret Sharing : Construction

An (n,n) secret sharing scheme for $s \in \{0,1\}$ based on XOR:

Share (s): sample random bits $(s_1, \ldots, s_n)$, s.t.,
$s_1 \oplus s_2 \oplus \ldots \oplus s_n = s$.

Reconstruct $(s_1', \ldots, s_n')$: Output $s_1' \oplus s_2' \oplus \ldots \oplus s_n'$

This is also known as the additive secret sharing scheme.

Security?

What if $s \in \mathbb{F}$?

Linearity of Additive Secret Sharing

Given additive secret shares $s_1 - - - - s_n$ of a secret s and additive secret shares $r_1 - - - r_n$ of a secret r, the parties can obtain secret shares of $u = s \oplus r$ as follows:

s_1, r_1

$u_1 = s_1 \oplus r_1$

s_2, r_2

$u_2 = s_2 \oplus r_2$

s_3, r_3

$u_3 = s_3 \oplus r_3$

s_4, r_4

$u_4 = s_4 \oplus r_4$

→ Does not require additional interaction

→ can compute shares of any linear function of s and r.

Secure Two-Party Computation of Linear Functions

	Alice		Bob
Input:	x		y
Function:		$\mathcal{L}$	
Protocol:	sample bits x^A, x^B s.t., $x^A \oplus x^B = x$	$\xrightarrow{x^B}$	sample bits y^A, y^B s.t., $y = y^A \oplus y^B$
		$\xleftarrow{y^A}$	
	compute $z^A = \mathcal{L}(x^A, y^A)$	$\xrightarrow{z^A}$	Compute $z^B = \mathcal{L}(x^B, y^B)$
		$\xleftarrow{z^B}$	
Output:	$z = z^A \oplus z^B$		$z = z^A \oplus z^B$

Secure Two-Party Computation of *General* Functions

Alice

Bob

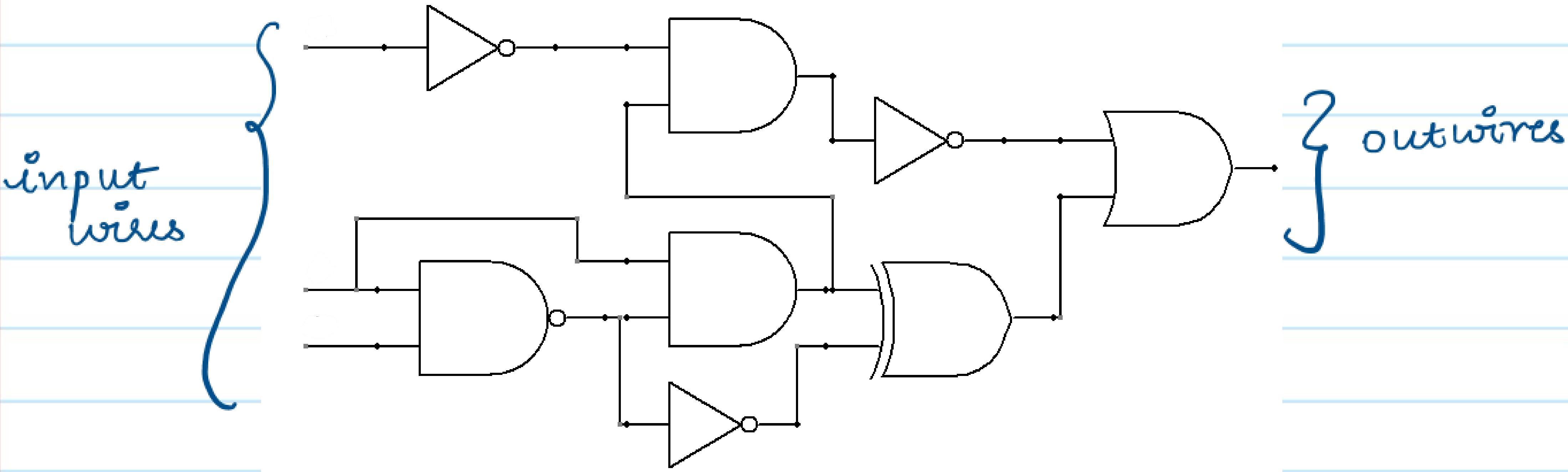
Input: $x_1, \ldots, x_m \in \{0,1\}^m$ $y_1, \ldots, y_m \in \{0,1\}^m$

Function: $f: \{0,1\}^{2m} \longrightarrow \{0,1\}^{\ell}$

Output: $f(x_1, \ldots, x_m, y_1, \ldots, y_m) = z_1, \ldots, z_\ell$

Function Representation

Function $f:\{0,1\}^{2m} \rightarrow \{0,1\}^{\ell}$ can be represented as a Boolean circuit:



each gate has fan-in at most 2.

Function Representation

Function $f:\{0,1\}^{2m} \rightarrow \{0,1\}^{l}$ can be represented as a Boolean circuit:

Input wires: $x_1, ---, x_m, y_1, --, y_m$

Output wires: $z_1, ---, z_l$

Gates: Since NAND gates are compute, we will assume that the circuit only comprises of AND and NOT gates.

GMW Protocol

for secure two-party computation of $f(a_1, \ldots a_m, b_1, \ldots b_m)$

Oded Goldreich

Silvio Micali

Avi Wigderson

Building Blocks

1. (2,2) Secret sharing
2. 1-out-of 4 Oblivious transfer

GMW Protocol

The invariant maintained throughout this protcol is that for every wire w in the circuit, Alice and Bob should have shares w_A, w_B, such that

$$\text{Reconstruct}(w_A, w_B) = w$$

When using additive secret sharing, this simply means

$$w_A \oplus w_B = w$$

3-Step Protocol:

1. Input Sharing
2. Circuit Evaluation
3. Output Reconstruction.

GMW Protocol : Input Sharing


Alice


Bob

Inputs : $x_1, \text{---}, x_m$ $y_1, \text{---}, y_m$

$\forall i \in [m]$: $\forall i \in [m]$:

$\text{Share}(x_i) \rightarrow x_i^A, x_i^B$ $\text{Share}(y_i) \rightarrow y_i^A, y_i^B$

$$\xrightarrow{x_1^B, \text{---}, x_m^B}$$

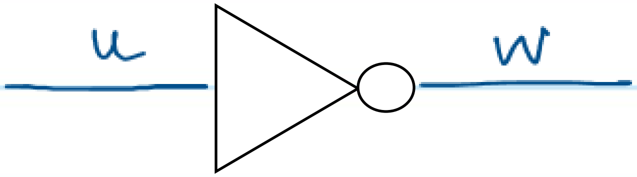
$$\xleftarrow{y_1^A, \text{---}, y_m^B}$$

GMW Protocol : Circuit Evaluation


Alice


Bob

NOT gate



Alice holds u^A

compute $w^A = u^A \oplus 1$

Bob holds u^B

compute $w^B = u^B$

Notice that $w^A \oplus w^B = u^A \oplus 1 \oplus u^B = \overline{u}$

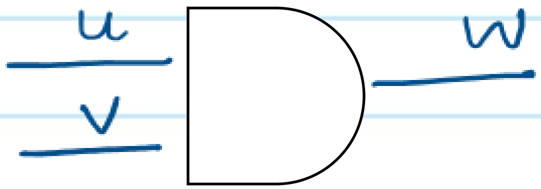
$\Rightarrow$ invariant is maintained !!

GMW Protocol : Circuit Evaluation


Alice


Bob

AND gate



Alice holds u^A, v^A

Bob holds u^B, v^B

Idea 1: Can they simply multiply their respective shares of u & v to obtain shares of w?

GMW Protocol : Circuit Evaluation

Alice

Bob

What do we want to compute? shares of w

what do we have? shares of u, v

$$w = u \cdot v \quad = \quad (u^A \oplus u^B)\cdot(v^A \oplus v^B)$$

$$= u^A \cdot v^A \oplus u^B \cdot v^B \oplus \underbrace{u^A \cdot v^B \oplus u^B \cdot v^A}_{\text{What about this?}}$$

Alice can compute this

Bob can compute this

GMW Protocol : Circuit Evaluation

How to compute shares of $u^A \cdot v^B + u^B \cdot v^A$?

Alice samples $r \xleftarrow{\$} \{0,1\}$

Alice's input to OT:

$a_{00} = r \oplus ((u^A \cdot 0) \oplus (v^A \cdot 0))$

$a_{01} = r \oplus ((u^A \cdot 1) \oplus (v^A \cdot 0))$

$a_{10} = r \oplus ((u^A \cdot 0) \oplus (v^A \cdot 1))$

$a_{11} = r \oplus ((u^A \cdot 1) \oplus (v^A \cdot 1))$

1-out-of-4 Oblivious transfer

Bob's input to OT:

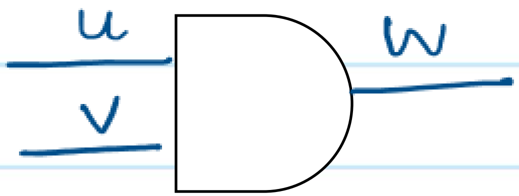
(u^B, v^B)

Alice sets her share of $u^A \cdot v^B + u^B \cdot v^A$ to be r

Bob sets his share of $u^A \cdot v^B + u^B \cdot v^A$ to be the output of OT

GMW Protocol : Circuit Evaluation

AND gate



- Alice holds u^A, v^A
- Sample $r \leftarrow \{0,1\}$ and use the following inputs to OT:

$$a_{00} = r \oplus ((u^A \cdot 0) \oplus (v^A \cdot 0))$$
$$a_{01} = r \oplus ((u^A \cdot 1) \oplus (v^A \cdot 0))$$
$$a_{10} = r \oplus ((u^A \cdot 0) \oplus (v^A \cdot 1))$$
$$a_{11} = r \oplus ((u^A \cdot 1) \oplus (v^A \cdot 1))$$

- Bob holds u^B, v^B
- use (u^B, v^B) as input the the OT protocol. Let s be the output of this OT.

- $w^A = u^A \cdot v^A \oplus r$
- $w^B = u^B \cdot v^B \oplus s$

Invariant is maintained!

GMW Protocol: Output Reconstruction

Alice

Bob

For all output wires: $z_1, \ldots, z_\ell$:

Alice holds

$z_1^A, \ldots, z_\ell^A$

Bob holds

$z_1^B, \ldots, z_\ell^B$

$$\xrightarrow{z_1^A, \ldots, z_\ell^A}$$

$$\xleftarrow{z_1^B, \ldots, z_\ell^B}$$

$\forall i \in [\ell]$

$z_i = z_i^A \oplus z_i^B$

$\forall i \in [\ell]$

$z_i = z_i^B \oplus z_i^A$