Lecture 9: Multiparty Computation

MIT - 6.893

Fall 2020

Henry Corrigan-Gibbs

Plan

* Multiparty computation
    - Concept
    - Applications
* Stretch break
* Simulation-based defn
* Breakout rooms

Logistics

- Unexpected Tuesday class next week
- Guest lecture next week from Susan MacGregor on privacy challenges for journalists.
- HW 3 due next Friday
- OH today

Multiparty computation

As far as I know, this idea goes back to Yao (1986)...

↳ Presented idea in conf talk, but not in paper

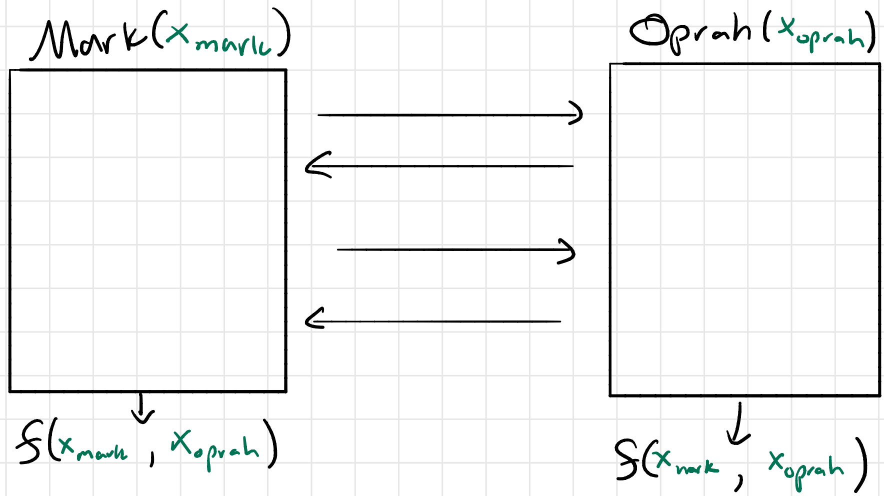
Historical aside: Yao is a legendary computer scientist

- Two PhDs: Physics & CS
- A number of papers that launched entire fields
  ↳ Comm complexity, cell-probe, MPC, ...
- It's worth checking out some of these papers
  e.g. "Should tables be sorted?"

Yao's "~~Millionaire's~~ Billionaire's Problem" (1986)

somewhere on a yacht in French polynesia....

- Each player i has private input $x_i \in \{0,1\}^\ell$
- Public function $f: \{0,1\}^\ell \times \{0,1\}^\ell \to \{0,1\}$



Intuitively, both parties learn $f(x_{mark}, x_{oprah})$, but they learn "nothing else" about the other's input

as always, devil is in the details

Multiparty Computation, Generally

Yao, GMW, ...

Parties $P_1, \ldots, P_n$

Each party P_i has a secret input $x_i \in \{0,1\}^\ell$

A public function $f : \underbrace{\{0,1\}^\ell \times \cdots \times \{0,1\}^\ell}_{n \text{ times}} \to \{0,1\}$

Goal: Jointly compute $f(x_1, \ldots, x_n)$ while "leaking nothing" to other parties about input.

A few things to notice:

- Easy to extend to many-bit output
  (Black-box: run many times in parallel)

- Each party can get private output

$$f(x_1, \ldots, x_n) \to (y_1, \ldots, y_n)$$

$$\hat{f}((x_1, r_1), \ldots, (x_n, r_n)) \to (y_1 + r_1, \ldots, x_n + r_n)$$

Private random blinding values

One-time pad encryption

MPC is powerful / general

PIR:

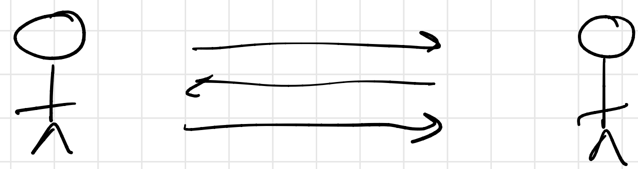
$$\text{Client}(i) \rightleftarrows \text{Server}(DB)$$

Client → DB_i Server → $\perp$

+ Restriction on comm complexity.

$$f(i, DB) := \{\text{output } i\text{th bit of } DB\}$$

Proofs:



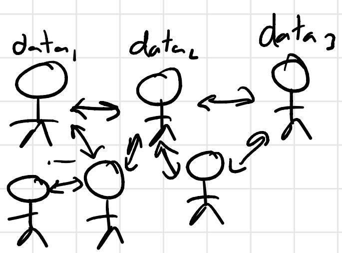
$N = p \cdot q$ Factors of N p, q

↓

"Yes, she knows factors of N"

$$f(N, p, q) = \{p \cdot q \overset{?}{=} N\}$$

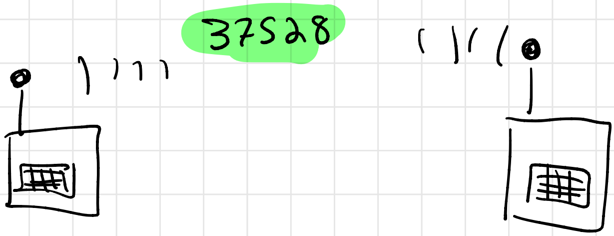
Privacy:



→ ML model trained on all of our data while leaking nothing else

$$f(data_1, \ldots, data_N) = \left\{ \begin{array}{l} \text{train model} \\ \text{on data} \end{array} \right\}$$

Relevant today: Exposure notification apps
(dramatically simplified!)



List of contacts

16284	Oct 6; 5:01 pm
56117	Oct 6; 6:04 pm
37528	Oct 7; 7:52 am
⋮	

There are serious problems with making this work in practice (Bluetooth, incentives, etc.) but I want to raise this example b/c it highlights both strengths + weaknesses of MPC.

- Every phone has:
  * a list of (ID, timestamp) pairs
  * a timestamp (if any) of when they tested positive for Covid.
- Every hour, every phone wants to know:

  "Did I have contact with other phone whose owner tested positive within a time window that would require me to quarantine?"

→ Learn "nothing else."

Types of MPC

You should remember: There are <u>many</u> types.

Things to specify:

- How many parties?
  ↳ Two, three, many?
  Interesting special cases for 2, 3.

- How many parties are adversarial?
  ... of n
  $< n/3$ "Byzantine" setting
  $< n/2$ "Honest majority"
  $< n-1$ "Dishonest majority"

- What type of misbehavior can adversary do?
  * collude + try to learn something while still following the protocol? "semi-honest"
  * arbitrarily deviate from the protocol "malicious"

- When does the adversary corrupt the parties?
  * Before protocol begins? "static"
  * While protocol runs? "adaptive"

- What type of security do we want?
  * No assumptions "info theoretic"
  * Crypto assumptions "computational"

- Do we want any "fairness" guarantee?
  * "fair" = all parties get output at same time
  * "sec w/ abort" = adv may get output, hon parties don't

Types of MPC

Not all combinations of these properties are feasible.

e.g. fairness requires an honest majority (Cleve 1986)

Good things to know:

Computational security:

GMW '87

GMW protocol: n parties, $\leq n-1$ malicious w/ abort (uses auth channels)

Yao's Garbled Ckts: 2 parties, better constants

Info theoretic:

BGW '88, CCD '88

BGW protocol: n parties $< n/3$ malicious (uses auth chanels)
or $< n/2$ malicious (with broadcast)

Complexity Metrics

* Computational costs for parties
* Communication (# bits exchanged)
* round complexity (# sequential msgs)

Problem: Most MPC protocols are not so concretely efficient.

↳ More when we see e.g. protocol.

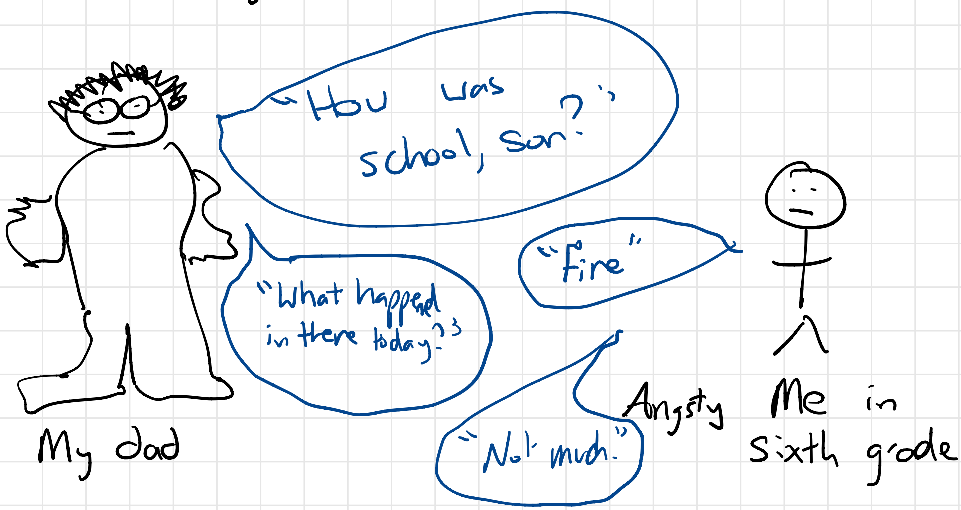
Stretch

Break

Key Idea: Simulation

The defn of MPC uses (arguably) the most beautiful idea in modern cryptography.
Using simulation to capture the idea of "learning nothing."

Idea: You have "learned nothing" from an interaction if you can write down a transcript of your interaction w/o actually interacting.

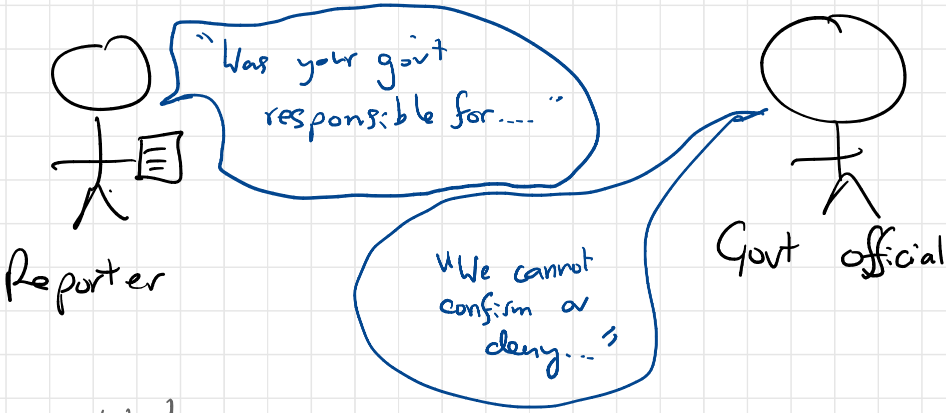


Might as well not have asked...

Dad can simulate a transcript of our interaction on his own...

Q: How was school?
A: Fire
Q: What happend...
A: Not much
⋮

Another example you see...



Can simulate!
Might as well not have asked!

Q: Have you been..
A: We cannot confirm...

[There's a life lesson here about how to ask good questions...]

In MPC context we want to capture the idea of "learning nothing" except $f(x_1, \dots, x_n)$

$\Longrightarrow$ Adversary should be able to simulate its "view" of the protocol given only $f(x_1, \dots, x_n)$.

"View" of a party in a protocol is
- input
- randomness
- msgs sent/received.

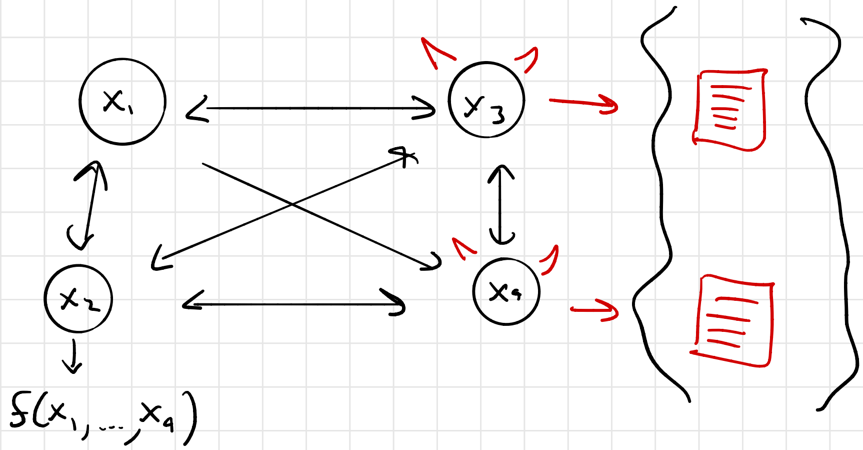
Definition

An MPC protocol is comp. secure against semi-honest advs if there exists an eff simulator Sim s.t. for every subset $C \subseteq [n]$ (e.g. $|C| < n/2$) and every choice of inputs $x_1, \dots, x_n$

$$\left\{ \begin{array}{c} \text{Views of} \\ \text{parties in } C \end{array} \right\} \overset{c}{\approx} \left\{ \mathrm{Sim}\left(\underbrace{\begin{array}{c} C, \{x_i \mid i \in C\}, \\ f(x_1, \dots, x_n) \end{array}}_{} \right) \right\}$$

Key idea: Input to simulator captures what leaks to adv.

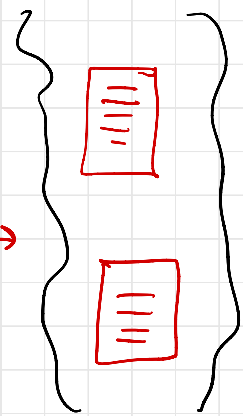
Real World



$$\stackrel{c}{\approx}$$

Ideal World

$$Sim\left(\{3,4\}, \{x_3, x_4\}, f(x_1, \ldots, x_4)\right) \rightarrow$$



Views capture everything adv learns during protocol execution...

Can simulate views $\Rightarrow$ "learns nothing except $f(x_1, \ldots, x_n)$"

Breakout Rooms

Definition gets more complicated

- in malicious model
- when $f(\cdot)$ is a stateful functionality
- when many instances of protocol may run concurrently.

Important Sanity checks!?

When you see a "privacy-preserving protocol" ask yourself:

* what flavor of MPC is at work? (type of adv, type of guarantee)
* What is the ideal functionality $f(\cdot)$ computing?
* If protocol leaks "nothing more than $f(x_1, \dots, x_n)$" is that good enough?

↳ EW example.