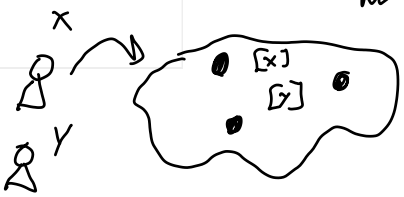
MPC

Monday, October 25, 2021 3:54 PM

Recap of MPC:

- n servers, up k of them can fail, we use secret sharing of degree k.



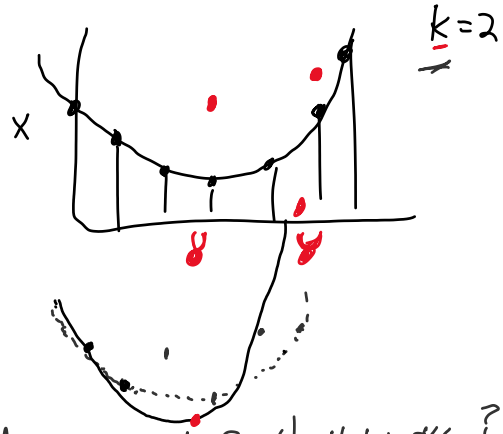
$a_0 = x \quad a_1 \xleftarrow{\$} \mathbb{F}_p \quad \cdots \quad a_k \xleftarrow{\$} \mathbb{F}_p$

- SecretShare(x): as client
  Sampling φ uniformly (among p^k possibilities)
  s.t. $\varphi(0) = x$ and φ is degree-k
  Send $\varphi(i)$ to server S_i as $[x]^{(i)}$

- Open($[x]$): as server S_i
  Send $[x]^{(i)}$ to each other server
  Receive n shares from other servers $[x]^{(j)}$
  Interpolate a polynomial φ s.t. $\varphi(j) = [x]^{(j)}$
  output $x = \varphi(0)$.

Robust Reconstruction.

- Suppose the servers that fail (up to k) they return invalid shares during Open
- Can we ensure we get the correct value anyway?



* Only one subset of 4 that works. ?

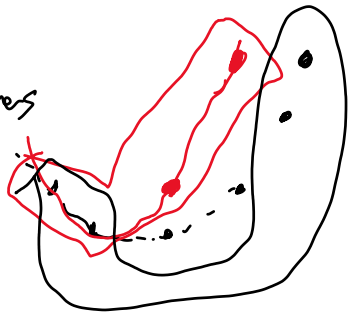
- Find a large enough subset. ~~4~~? 5?
- Potentially two subsets each intersecting 4 points

- If we can find a $\deg k$ poly intersecting

2K+1 received shares.
⇒ we can conclude it is the correct one.

- Among 2K+1 received shares, at most K come from malicious nodes, so at least K+1 are correct shares. And K+1 correct shares uniquely determine the correct φ.

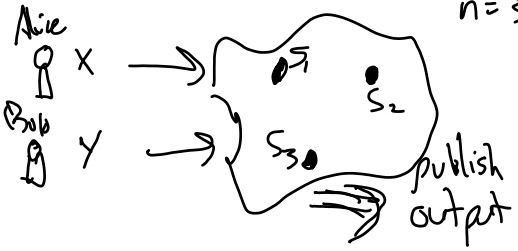
- $n \geq 2K+1$ is the minimum number of servers relative to fault tolerance K for which robust interpolation works.

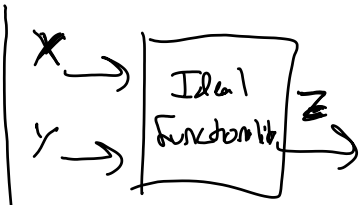


- $n \geq 3K+1$ is the minimum to ensure guaranteed output
(if the 2K+1 first shares aren't successful, wait for more and try again.
Output once some subset of 2K+1 received shares lie on a deg K poly.).

Simulation Security for MPC.

$n=3$, $K=1$, semi-honest





View can be simulated given just Z

Input: X, Y
Preproc: [a], [b], [ab].
Procedure:
Alice secretshares(x)
Bob secretshares(y)
Servers:
D := Open([x]-[a])
E := Open([y]-[b])
[z] := DE + .. + [ab]
output z := Open([z])

* Alt interpretation:
flip a bit, either produce the real view or the simulation.
These are indistinguishable.

* for all X_L, Y_L, X_R, Y_R, s.t. $Y_L \cdot X_L = Y_R \cdot X_R$.
$View_L \approx View_R$

- What is the view of S_1 in this protocol? Beaver mul
enumerate everything S_1 sees: a

Server i sees: $[x]^{(i)}, [y]^{(i)}$.

- $[a], [b], [ab]$
- φ_D from reconstruction
- φ_E
- φ_Z

Total # p.
$2x$
$3x$
$2x$ (deg | poly)
$\varphi_D(i) = [x \cdot a]$ $2x$
$2x$

Total $11x$ Fp elems.

open($[D]$) $[D]^{(i)}$

All these $[D]$ from deg. 1 line.

$[D]^{(3)}$ $[D]^{(2)}$

- Give a Pr distribution

$\varphi_D(i) = [x \cdot a]$ DoF -1
$\varphi_E(i) = [y \cdot b]$ -1
$\varphi_Z(i) = DF + \cdots [ab]$ -1
$\varphi_Z(0) = z$ -1

In total, $p^{\vec{}}$ possible views, all equally likely.

- How can it be simulated given z.

given prob dist. above, sample in any order that satisfies constraints.

$S(z)$:

$\varphi_Z \overset{\$}{\leftarrow}$ deg-1 poly.

Constant Round Vector Product.

Input: $[x_1] \ldots [x_m]$ Precondition: $x_i \neq 0$.

Output: $[x_1 \cdot x_2 \cdots x_m] = [\prod x_i]$

Preprocessi. any.

Perf goal: constant # of rounds

$[x]$ $[y]$

X

$[xy]$

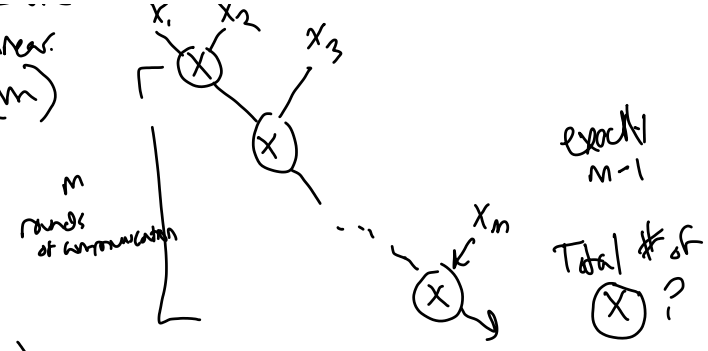
Initial: $\Delta^x \rightarrow [x]$

- open(D) · open(E)

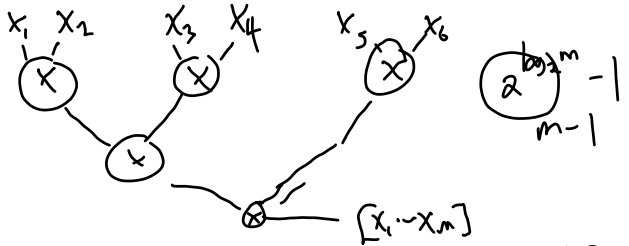
output $\rightarrow [xy] \rightarrow xy$

Procedure: ...

linear.
$O(m)$



$O(\log m)$:



$O(1)$: lookup table? and overhead is $O(m)$.

related: $[x_1], \cdots [x_k]$

filtration $[:]$ output: $[x_i]$

$$\sum_{j=1}^{k} \left([:] \stackrel{?}{=} j\right) \cdot [x_j]$$

$x_1 \times x_2 \times \cdots x_m$

"mask" $D = (x-a)$ ← 2x secure mults

$r_0, r_1, \cdots r_m \xleftarrow{\$} \mathbb{F}_p$ $\quad [r_1] \times [x_2] \times [r_2^{-1}]$

$\Downarrow$ open$([r_1 x_2 r_2^{-1}])$

$$(r_0 \cdot x_1 \cdot r_1^{-1}) \quad (r_1 \cdot x_2 \cdot r_2^{-1}) \cdots (r_{m-1} x_m r_m^{-1})$$

$x_1 \neq 0 \Rightarrow (r_0 \cdot x_1 \cdot r_1^{-1})$ uniformly distributed.

all $r_i x_{i+1} r_{i+1}^{-1}$ uniformly distributed

$$[r_0^{-1}] \times \left(\prod_{i=0}^{m} (r_i x_i r_i^{-1}) \times [r_m] \right) = [x_1 \cdots x_m]$$

- possible to do w/ only $r_1 \cdots r_m$? Choose $[r_m]$ s.t. $r_m \cdot \prod_{i<m} r_i = 1$.
- precomp $[r_i]\,[r_i^{-1}]$ + 2x secure triples
- Total cost: ~ 2.5x overhead?

DFT over finite fields.

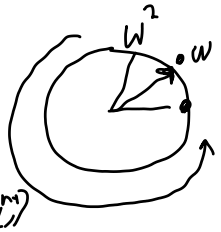
DFT (over reals)

$$X_k = \sum_{j=0}^{n-1} x_j \cdot e^{(2\pi i/n) \cdot k \cdot j} \qquad \sum x_j \omega^{kj}$$

Primitive n'th roots of unity

- n'th root of unity is $\omega^n = 1$
- primitive $\Rightarrow$ n is the smallest > 0 for which is true.

$$\omega = e^{2\pi i / n}$$



time $\longleftrightarrow$ freq.

coeff $\longleftrightarrow$ eval-point. (w/ basis $\omega_1 \ldots \omega^{n-1}$)

$$DFT_\omega[\varphi] = \{\varphi(0), \varphi(\omega), \varphi(\omega^2), \ldots \varphi(\omega^{n-1})\}$$

$$\varphi(\omega^j) = a_0 + a_1(\omega^j) + a_2(\omega^{j \cdot 2}) \ldots a_{n-1}(\omega^{j \cdot (n-1)})$$

- This can be computed in $O(n \log n)$
  - n is a power of 2.
- finding ω (prim. nth root of unity?) $\mathbb{F}_p$
  - Check if ω is an nth root of unity? $\omega^n \stackrel{?}{=} 1$
  - Lagrange $\Rightarrow$ ω is an n'th root of unity

$n \mid p-1$

ex. $p = (2^k \cdot q) + 1$ $\quad n = 2^k$