Scalable Multiparty Computation from Non-linear Secret Sharing

Sanjam Garg

UC Berkeley

Abhishek Jain

JHU & NTT Research

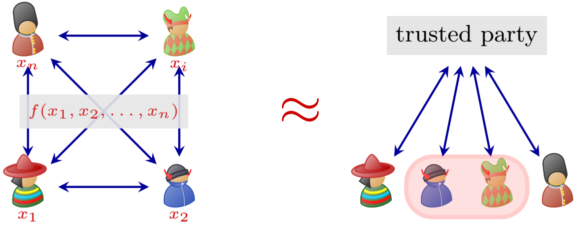
Pratyay Mukherjee

Supra Research

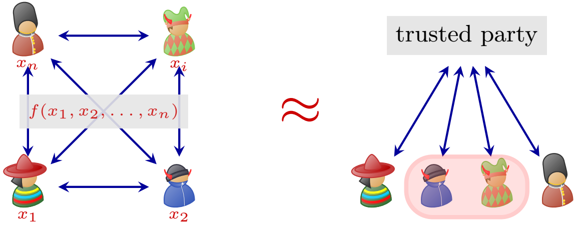
Mingyuan Wang

UC Berkeley -> NYU Shanghai

Aug. 2024 @ CRYPTO

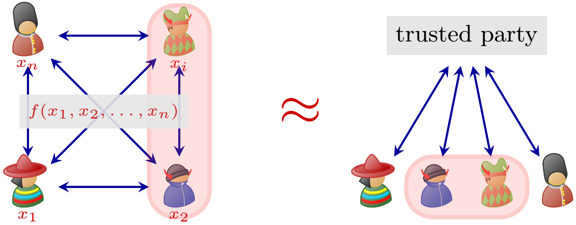


x_n
x_i
$f(x_1, x_2, \ldots, x_n)$
x_1
x_2
$\approx$
trusted party



This Work

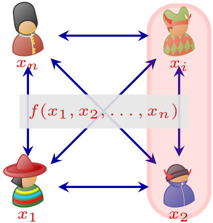
- honest majority
- information-theoretic plain model
- semi-honest adversary



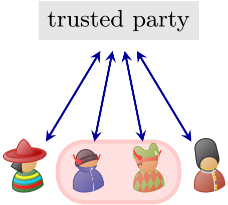
This Work

- honest majority
- information-theoretic plain model
- semi-honest adversary

Objective



$\approx$

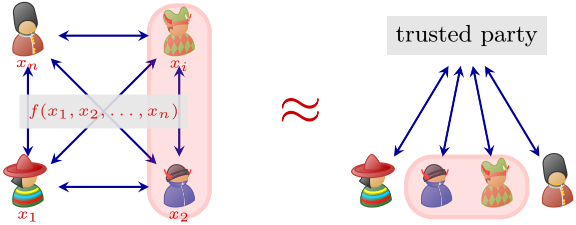


This Work

- honest majority
- information-theoretic plain model
- semi-honest adversary

Objective

- Minimizing overall communication & computation complexity

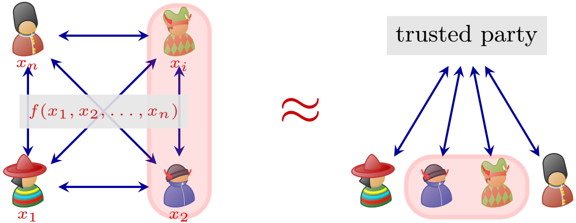


This Work

- honest majority
- information-theoretic plain model
- semi-honest adversary

Objective

- Minimizing overall communication & computation complexity
- For an arithmetic circuit C over F, can we achieve overall computation complexity $|C|$ field operations?
  - Optimal since insecure evaluation requires the same complexity



This Work

- honest majority
- information-theoretic plain model
- semi-honest adversary

Objective

- Minimizing overall communication & computation complexity
- For an arithmetic circuit C over F, can we achieve overall computation complexity $|C|$ field operations?
  - Optimal since insecure evaluation requires the same complexity
- Scalable as the overall complexity does not grow with n.

Why Scalable information-theoretic MPC

- Most practically-efficient MPC protocols (only field operations, no cryptographic operations)

Why Scalable information-theoretic MPC

- Most practically-efficient MPC protocols (only field operations, no cryptographic operations)
- Can distribute a large computation over parties; per party workload decreases as n grows

Why Scalable information-theoretic MPC

- Most practically-efficient MPC protocols (only field operations, no cryptographic operations)
- Can distribute a large computation over parties; per party workload decreases as n grows
- Honest majority assumption becomes more reliable as n grows

Why Scalable information-theoretic MPC

- Most practically-efficient MPC protocols (only field operations, no cryptographic operations)
- Can distribute a large computation over parties; per party workload decreases as n grows
- Honest majority assumption becomes more reliable as n grows
- Many applications naturally involve many parties (e.g., federated learning)

Why Scalable information-theoretic MPC

- Most practically-efficient MPC protocols (only field operations, no cryptographic operations)
- Can distribute a large computation over parties; per party workload decreases as n grows
- Honest majority assumption becomes more reliable as n grows
- Many applications naturally involve many parties (e.g., federated learning)
- ...

Prior Works

After a long sequence of works [Ben-Or-Goldwasser-Wigderson'88, Chaum-Crepeau-Damgard'88, Franklin-Yung'92, Damgard-Nielson'07, ...]

Prior Works

After a long sequence of works [Ben-Or-Goldwasser-Wigderson'88, Chaum-Crepeau-Damgard'88, Franklin-Yung'92, Damgard-Nielson'07, ...]

Structured Circuit

- SIMD circuit [Franklin-Yung'92]
- highly-repeatitive circuit [Beck-Goel-Jain-Kaptchuk Eurocrypt'21]

Prior Works

After a long sequence of works [Ben-Or-Goldwasser-Wigderson'88, Chaum-Crepeau-Damgard'88, Franklin-Yung'92, Damgard-Nielson'07, ...]

Structured Circuit

- SIMD circuit [Franklin-Yung'92]
- highly-repeatitive circuit [Beck-Goel-Jain-Kaptchuk Eurocrypt'21]

General Circuit

- Circuit transformation [Damgard-Ishai-Kroigaard'10, Genkin-Ishai-Polychroniadou'15]
  - introduces $\mathsf{poly}(\log|C|, d)$ overhead (communication/computation, round complexity)

Prior Works

After a long sequence of works [Ben-Or-Goldwasser-Wigderson'88, Chaum-Crepeau-Damgard'88, Franklin-Yung'92, Damgard-Nielson'07, ...]

Structured Circuit

- SIMD circuit [Franklin-Yung'92]
- highly-repeatitive circuit [Beck-Goel-Jain-Kaptchuk Eurocrypt'21]

General Circuit

- Circuit transformation [Damgard-Ishai-Kroigaard'10, Genkin-Ishai-Polychroniadou'15]
  - introduces $\mathsf{poly}(\log |C|, d)$ overhead (communication/computation, round complexity)
- Share transformation [Goyal-Polychroniadou-Song'21, Goyal-Polychroniadou-Song'22]
  - Only achieve communication complexity $|C|$ field element
  - Computation complexity is still $n \cdot |C|$ field operation

Can we build scalable MPC protocol in computation for general circuit?

Our Results (Informal)

Assuming F is an exponentially large prime field,

For any general circuit C over F, there is a scalable MPC protocol among n parties

Our Results (Informal)

Assuming F is an exponentially large prime field,

For any general circuit C over F, there is a scalable MPC protocol among n parties

- The communication/computation (bit)-complexity is $O(|C| \cdot \log |F|)$ ($\approx |C|$ field elements/operations)

Our Results (Informal)

Assuming F is an exponentially large prime field,

- We prove security when $\log|F| = \widetilde{O}(n^2)$

For any general circuit C over F, there is a scalable MPC protocol among n parties

- The communication/computation (bit)-complexity is $O(|C| \cdot \log|F|)$ ($\approx |C|$ field elements/operations)

Our Results (Informal)

Assuming F is an exponentially large prime field,

- We prove security when $\log|F| = \widetilde{O}(n^2)$
- We conjecture it is secure even when $\log|F| = \widetilde{O}(n)$

For any general circuit C over F, there is a scalable MPC protocol among n parties

- The communication/computation (bit)-complexity is $O(|C| \cdot \log|F|)$ ($\approx |C|$ field elements/operations)

Our Results (Informal)

Assuming F is an exponentially large prime field,

- We prove security when $\log|F| = \widetilde{O}(n^2)$
- We conjecture it is secure even when $\log|F| = \widetilde{O}(n)$

For any general circuit C over F, there is a scalable MPC protocol among n parties

- The communication/computation (bit)-complexity is $O(|C| \cdot \log|F|)$ ($\approx |C|$ field elements/operations)

- We measure complexity at a bit-level (as opposed to $|C|$ field operations)

Our Results (Informal)

Assuming F is an exponentially large prime field,

- We prove security when $\log|F| = \widetilde{O}(n^2)$
- We conjecture it is secure even when $\log|F| = \widetilde{O}(n)$

For any general circuit C over F, there is a scalable MPC protocol among n parties

- The communication/computation (bit)-complexity is $O(|C| \cdot \log|F|)$ ($\approx |C|$ field elements/operations)

- We measure complexity at a bit-level (as opposed to $|C|$ field operations)
- Also extends to dishonest-majority setting in the preprocessing model (see paper)

Our Results (Informal)

Assuming F is an exponentially large prime field,

- We prove security when $\log |F| = \widetilde{O}(n^2)$
- We conjecture it is secure even when $\log |F| = \widetilde{O}(n)$

For any general circuit C over F, there is a scalable MPC protocol among n parties

- The communication/computation (bit)-complexity is $O(|C| \cdot \log |F|)$ ($\approx |C|$ field elements/operations)

- We measure complexity at a bit-level (as opposed to $|C|$ field operations)
- Also extends to dishonest-majority setting in the preprocessing model (see paper)

An alternative approach from existing works

- Translate the arithmetic circuit into a Boolean circuit $\Longrightarrow$ highly-repeatitive (boolean) circuit [Beck-Goel-Jain-Kaptchuk Eurocrypt'21]

Our Results (Informal)

Assuming F is an exponentially large prime field,

- We prove security when $\log|F| = \widetilde{O}(n^2)$
- We conjecture it is secure even when $\log|F| = \widetilde{O}(n)$

For any general circuit C over F, there is a scalable MPC protocol among n parties

- The communication/computation (bit)-complexity is $O(|C| \cdot \log|F|)$ ($\approx |C|$ field elements/operations)

- We measure complexity at a bit-level (as opposed to $|C|$ field operations)
- Also extends to dishonest-majority setting in the preprocessing model (see paper)

An alternative approach from existing works

- Translate the arithmetic circuit into a Boolean circuit $\Longrightarrow$ highly-repeatitive (boolean) circuit [Beck-Goel-Jain-Kaptchuk Eurocrypt'21]
- Not desirable due to high (concrete/asymptotic) cost of computation
  - Yao's Garbling vs. Arithmetic Garbling: [Applebaum-Ishai-Kushilevitz'11]

Application of MPC over large prime field

Delegating computation of resource-intensive cryptographic tasks:

- SNARK proof generation [Ozdemir-Boneh'22, Garg-Goel-Jain-Policharla-Sekar'23, Chiesa-Lehmkuhl-Mishra-Zhang'23]
- $\log |F| \approx 256$

Application of MPC over large prime field

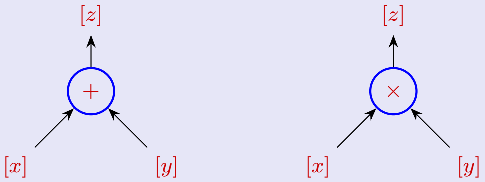
Delegating computation of resource-intensive cryptographic tasks:

- SNARK proof generation [Ozdemir-Boneh'22, Garg-Goel-Jain-Policharla-Sekar'23, Chiesa-Lehmkuhl-Mishra-Zhang'23]
- $\log|F| \approx 256$
- Our protocol can plausibly ($n < \log|F|$) be applied to such scenarios with $100 \sim 200$ parties.

Technical Highlight

Existing Framework

Emulating the circuit evaluation gate by gate by secret sharing

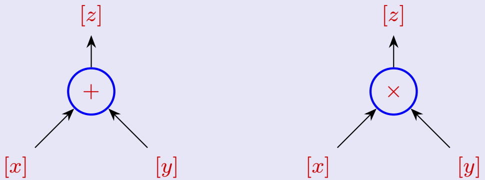


Tricks required for Scalable MPC

- Packed secret sharing [Franklin-Yung'92]
  - This work: "Unpacked" secret sharing
- Batch Randomness Generation via VanderMonde randomness extraction [Damgard-Nielson'07]
  - This work: High-dimensional Sampling Lemma

Existing Framework

Emulating the circuit evaluation gate by gate by secret sharing



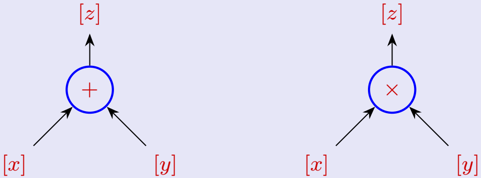
- Given $[x]$ and $[y]$, locally compute $[z] = [x] + [y]$ or $[x] \cdot [y]$

Tricks required for Scalable MPC

- Packed secret sharing [Franklin-Yung'92]
  - This work: "Unpacked" secret sharing
- Batch Randomness Generation via VanderMonde randomness extraction [Damgard-Nielson'07]
  - This work: High-dimensional Sampling Lemma

Existing Framework

Emulating the circuit evaluation gate by gate by secret sharing



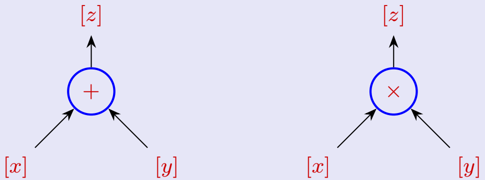
- Given $[x]$ and $[y]$, locally compute $[z] = [x] + [y]$ or $[x] \cdot [y]$
- Degree-reduction after each multiplication gate, given double sharing $[r]_t$ and $[r]_{2t}$ of r
  - Reconstruct $[x]_t \cdot [y]_t - [r]_{2t}$
  - Locally compute $[z] = [r]_t + (x \cdot y - r)$

Tricks required for Scalable MPC

- Packed secret sharing [Franklin-Yung'92]
  - This work: "Unpacked" secret sharing
- Batch Randomness Generation via VanderMonde randomness extraction [Damgard-Nielson'07]
  - This work: High-dimensional Smudging Lemma

Existing Framework

Emulating the circuit evaluation gate by gate by secret sharing



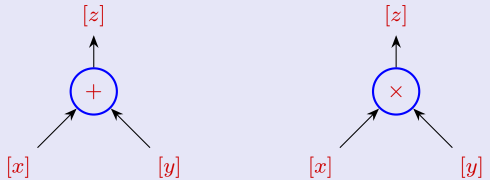
- Given $[x]$ and $[y]$, locally compute $[z] = [x] + [y]$ or $[x] \cdot [y]$
- Degree-reduction after each multiplication gate, given double sharing $[r]_t$ and $[r]_{2t}$ of r
  - Reconstruct $[x]_t \cdot [y]_t - [r]_{2t}$
  - Locally compute $[z] = [r]_t + (x \cdot y - r)$

Tricks required for Scalable MPC

- Packed secret sharing [Franklin-Yung'92]
  - This work: "Unpacked" secret sharing
- Batch Randomness Generation via VanderMonde randomness extraction [Damgard-Nielson'07]
  - This work: High-dimensional Smudging Lemma

Existing Framework

Emulating the circuit evaluation gate by gate by secret sharing



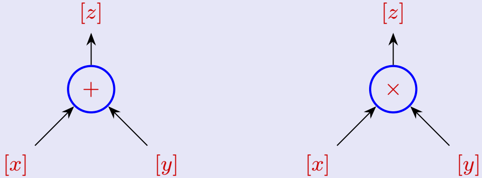
- Given $[x]$ and $[y]$, locally compute $[z] = [x] + [y]$ or $[x] \cdot [y]$
- Degree-reduction after each multiplication gate, given double sharing $[r]_t$ and $[r]_{2t}$ of r
  - Reconstruct $[x]_t \cdot [y]_t - [r]_{2t}$
  - Locally compute $[z] = [r]_t + (x \cdot y - r)$

Tricks required for Scalable MPC

- Packed secret sharing [Franklin-Yung'92]
  - This work: "Unpacked" secret sharing
- Batch Randomness Generation via VanderMonde randomness extraction [Damgard-Nielson'07]
  - This work: High-dimensional Smudging Lemma

Existing Framework

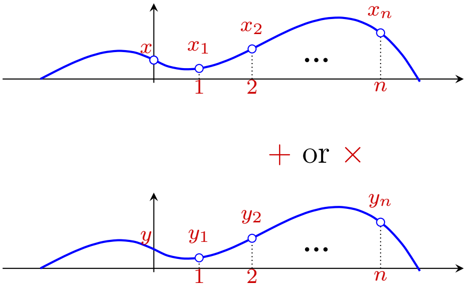
Emulating the circuit evaluation gate by gate by secret sharing



- Given $[x]$ and $[y]$, locally compute $[z] = [x] + [y]$ or $[x] \cdot [y]$
- Degree-reduction after each multiplication gate, given double sharing $[r]_t$ and $[r]_{2t}$ of r
  - Reconstruct $[x]_t \cdot [y]_t - [r]_{2t}$
  - Locally compute $[z] = [r]_t + (x \cdot y - r)$

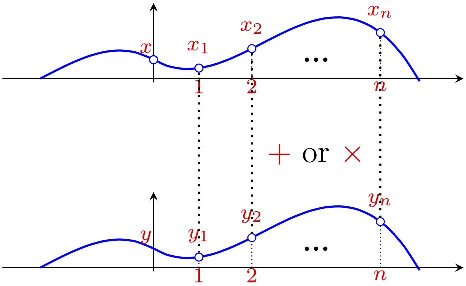
Tricks required for Scalable MPC

- Packed secret sharing [Franklin-Yung'92]
  - This work: "Unpacked" secret sharing
- Batch Randomness Generation via VanderMonde randomness extraction [Damgard-Nielson'07]
  - This work: High-dimensional Smudging Lemma



Necessity of Packing

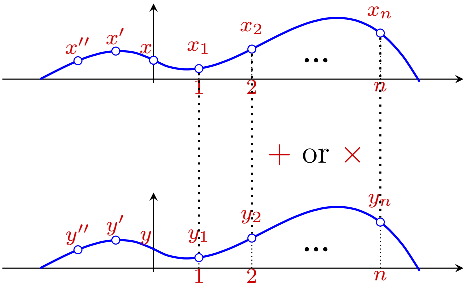
Limitations of Packing



Necessity of Packing

- n field operations for emulating one arithmetic gate
  - This is the case for any linear secret sharing scheme

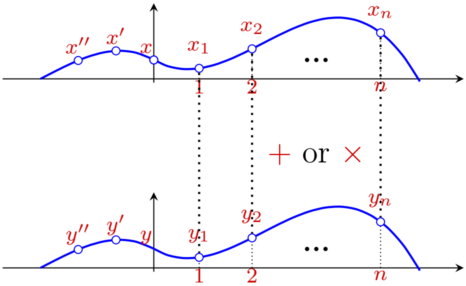
Limitations of Packing



Necessity of Packing

- n field operations for emulating one arithmetic gate
  - This is the case for any linear secret sharing scheme
- Packing $O(n)$ secrets into one instance of a secret sharing
  - $O(n)$ overhead becomes $O(1)$ through packing

Limitations of Packing

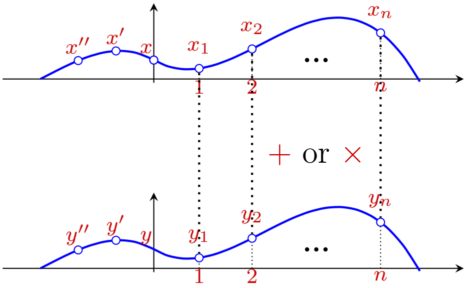


Necessity of Packing

- n field operations for emulating one arithmetic gate
  - This is the case for any linear secret sharing scheme
- Packing $O(n)$ secrets into one instance of a secret sharing
  - $O(n)$ overhead becomes $O(1)$ through packing

Limitations of Packing

- Must emulate multiple $O(n)$ gates simultaneously

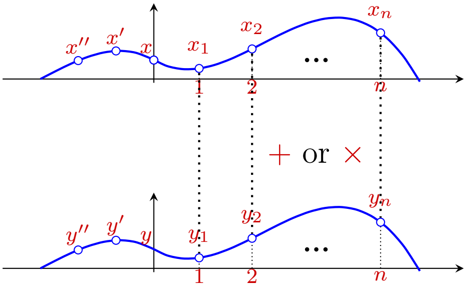


Necessity of Packing

- n field operations for emulating one arithmetic gate
  - This is the case for any linear secret sharing scheme
- Packing $O(n)$ secrets into one instance of a secret sharing
  - $O(n)$ overhead becomes $O(1)$ through packing

Limitations of Packing

- Must emulate multiple $O(n)$ gates simultaneously
- Existing works develop different ways to tackle this
  - Structure circuit / circuit transformation [Franklin-Yung'92, Damgard-Ishai-Kroigaard'10, Genkin-Ishai-Polychroniadou'15]
  - Share transformation [Goyal-Polychroniadou-Song'21, Goyal-Polychroniadou-Song'22]

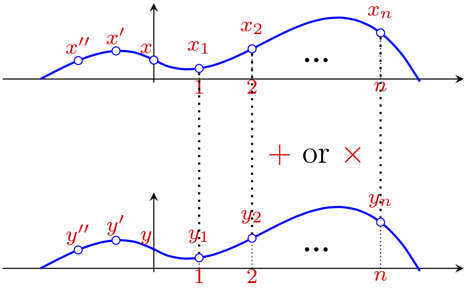


Necessity of Packing

- n field operations for emulating one arithmetic gate
  - This is the case for any linear secret sharing scheme
- Packing $O(n)$ secrets into one instance of a secret sharing
  - $O(n)$ overhead becomes $O(1)$ through packing

Limitations of Packing

- Must emulate multiple $O(n)$ gates simultaneously
- Existing works develop different ways to tackle this
  - Structure circuit / circuit transformation [Franklin-Yung'92, Damgard-Ishai-Kroigaard'10, Genkin-Ishai-Polychroniadou'15]
  - Share transformation [Goyal-Polychroniadou-Song'21, Goyal-Polychroniadou-Song'22]
- Can't achieve computational scalability for general circuit



Necessity of Packing

- n field operations for emulating one arithmetic gate
  - This is the case for any linear secret sharing scheme
- Packing $O(n)$ secrets into one instance of a secret sharing
  - $O(n)$ overhead becomes $O(1)$ through packing

Limitations of Packing

- Must emulate multiple $O(n)$ gates simultaneously
- Existing works develop different ways to tackle this
  - Structure circuit / circuit transformation [Franklin-Yung'92, Damgard-Ishai-Kroigaard'10, Genkin-Ishai-Polychroniadou'15]
  - Share transformation [Goyal-Polychroniadou-Song'21, Goyal-Polychroniadou-Song'22]
- Can't achieve computational scalability for general circuit

Key Point of Packing

- Efficiency: rate $O(1)$ secret sharing through packing.
- Drawback: *amortized* rate

Key Point of Packing

- Efficiency: rate $O(1)$ secret sharing through packing.
- Drawback: *amortized* rate

Key Point of Packing

- Efficiency: rate $O(1)$ secret sharing through packing.
- Drawback: *amortized* rate

Our Conceptual Contribution

Achieving *non-amortized* rate $O(1)$ secret sharing through “unpacking”.

Key Point of Packing

- Efficiency: rate $O(1)$ secret sharing through packing.
- Drawback: *amortized* rate

Our Conceptual Contribution

Achieving *non-amortized* rate $O(1)$ secret sharing through "unpacking".

- Breaking long secret into short secret shares

Key Point of Packing

- Efficiency: rate $O(1)$ secret sharing through packing.
- Drawback: *amortized* rate

Our Conceptual Contribution

Achieving *non-amortized* rate $O(1)$ secret sharing through "unpacking".

- Breaking long secret into short secret shares
- No need to emulate multiple gates simultaneously

Key Point of Packing

- Efficiency: rate $O(1)$ secret sharing through packing.
- Drawback: *amortized* rate

Our Conceptual Contribution

Achieving *non-amortized* rate $O(1)$ secret sharing through "unpacking".

- Breaking long secret into short secret shares
- No need to emulate multiple gates simultaneously

How can we achieve this?

Chinese-remainder-Theorem based Secret Sharing

- recently introduced by [GJMSWZ'23] to build (weighted) mpc protocols
  - compatible with the existing framework; gate emulation + degree reduction

Remarks

Chinese-remainder-Theorem based Secret Sharing

- recently introduced by [GJMSWZ'23] to build (weighted) mpc protocols
  - compatible with the existing framework; gate emulation + degree reduction
- A secret $s \in F_p$ is re-randomized as an integer $S = s + \alpha \cdot p$.

Remarks

Chinese-remainder-Theorem based Secret Sharing

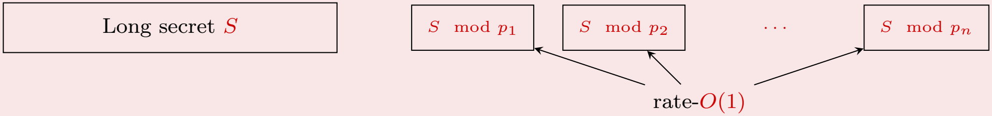
- recently introduced by [GJMSWZ'23] to build (weighted) mpc protocols
  - compatible with the existing framework; gate emulation + degree reduction
- A secret $s \in F_p$ is re-randomized as an integer $S = s + \alpha \cdot p$.

Long secret S	$S \mod p_1$	$S \mod p_2$	$\cdots$	$S \mod p_n$

Remarks

Chinese-remainder-Theorem based Secret Sharing

- recently introduced by [GJMSWZ'23] to build (weighted) mpc protocols
  - compatible with the existing framework; gate emulation + degree reduction
- A secret $s \in F_p$ is re-randomized as an integer $S = s + \alpha \cdot p$.

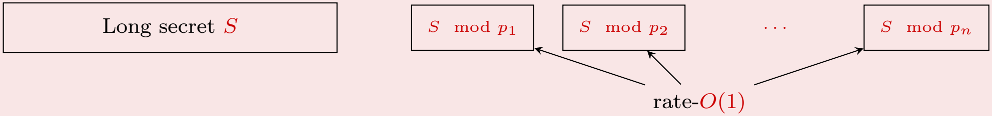


- Each share can be much smaller than the secret (e.g., $p_1 = 2$, $p_2 = 3$, ...)
- Pick $p_1, p_2, \ldots, p_n$ appropriately to make it rate-$O(1)$.

Remarks

Chinese-remainder-Theorem based Secret Sharing

- recently introduced by [GJMSWZ'23] to build (weighted) mpc protocols
  - compatible with the existing framework; gate emulation + degree reduction
- A secret $s \in F_p$ is re-randomized as an integer $S = s + \alpha \cdot p$.



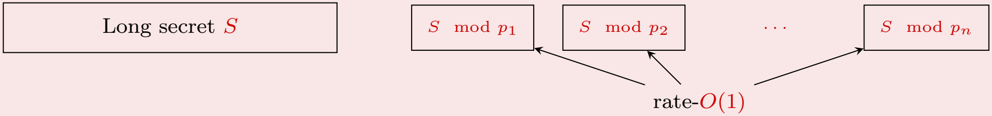
- Each share can be much smaller than the secret (e.g., $p_1 = 2$, $p_2 = 3$, ...)
- Pick $p_1, p_2, \ldots, p_n$ appropriately to make it rate-$O(1)$.

Remarks

- Secret length $\log F$ has to be $O(n)$

Chinese-remainder-Theorem based Secret Sharing

- recently introduced by [GJMSWZ'23] to build (weighted) mpc protocols
  - compatible with the existing framework; gate emulation + degree reduction
- A secret $s \in F_p$ is re-randomized as an integer $S = s + \alpha \cdot p$.



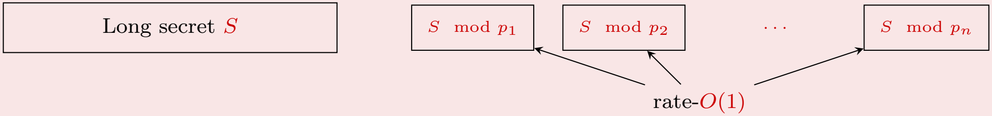
- Each share can be much smaller than the secret (e.g., $p_1 = 2$, $p_2 = 3$, ...)
- Pick $p_1, p_2, \ldots, p_n$ appropriately to make it rate-$O(1)$.

Remarks

- Secret length $\log F$ has to be $O(n)$
- Have to measure overall complexity at a bit level

Chinese-remainder-Theorem based Secret Sharing

- recently introduced by [GJMSWZ'23] to build (weighted) mpc protocols
  - compatible with the existing framework; gate emulation + degree reduction
- A secret $s \in F_p$ is re-randomized as an integer $S = s + \alpha \cdot p$.



- Each share can be much smaller than the secret (e.g., $p_1 = 2$, $p_2 = 3$, ...)
- Pick $p_1, p_2, \ldots, p_n$ appropriately to make it rate-$O(1)$.

Remarks

- Secret length $\log F$ has to be $O(n)$
- Have to measure overall complexity at a bit level
- Already achieve online overhead $O(1)$ assuming we have $[r]_t$ and $[r]_{2t}$

Batch Randomness Generation

How do we generate $[r]_t$, $[r]_{2t}$ efficiently?

Batch Randomness Generation

How do we generate $[r]_t$, $[r]_{2t}$ efficiently?

- Each multiplication gate consumes one pair;

Batch Randomness Generation

How do we generate $[r]_t$, $[r]_{2t}$ efficiently?

- Each multiplication gate consumes one pair;
- Each pair should be generated with complexity not dependent on n

Batch Randomness Generation

How do we generate $[r]_t$, $[r]_{2t}$ efficiently?

- Each multiplication gate consumes one pair;
- Each pair should be generated with complexity not dependent on n

VanderMonde Randomness Extraction Damgard-Nielson'07

Each party generates a pair $[r_i]_t$, $[r_i]_{2t}$ and

$$V_{n-t,n} \cdot \begin{pmatrix} [r_1]_t \\ [r_2]_t \\ \vdots \\ [r_n]_t \end{pmatrix} \qquad V_{n-t,n} \cdot \begin{pmatrix} [r_1]_{2t} \\ [r_2]_{2t} \\ \vdots \\ [r_n]_{2t} \end{pmatrix}, \qquad \text{e.g.,} \quad V_{n-t,n} = \begin{pmatrix} 1 & 2 & 3 & \dots & n \\ 1^2 & 2^2 & 3^2 & \dots & n^2 \\ \\ 1^{n-t} & 2^{n-t} & 3^{n-t} & \dots & n^{n-t} \end{pmatrix}$$

Batch Randomness Generation

How do we generate $[r]_t$, $[r]_{2t}$ efficiently?

- Each multiplication gate consumes one pair;
- Each pair should be generated with complexity not dependent on n

VanderMonde Randomness Extraction Damgard-Nielson'07

Each party generates a pair $[r_i]_t$, $[r_i]_{2t}$ and

$$V_{n-t,n} \cdot \begin{pmatrix} [r_1]_t \\ [r_2]_t \\ \vdots \\ [r_n]_t \end{pmatrix} \qquad V_{n-t,n} \cdot \begin{pmatrix} [r_1]_{2t} \\ [r_2]_{2t} \\ \vdots \\ [r_n]_{2t} \end{pmatrix}, \qquad \text{e.g.,} \quad V_{n-t,n} = \begin{pmatrix} 1 & 2 & 3 & \dots & n \\ 1^2 & 2^2 & 3^2 & \dots & n^2 \\ & & & & \\ 1^{n-t} & 2^{n-t} & 3^{n-t} & \dots & n^{n-t} \end{pmatrix}$$

- Extract $n-t$ pairs out of n pairs

Batch Randomness Generation

How do we generate $[r]_t$, $[r]_{2t}$ efficiently?

- Each multiplication gate consumes one pair;
- Each pair should be generated with complexity not dependent on n

VanderMonde Randomness Extraction Damgard-Nielson'07

Each party generates a pair $[r_i]_t$, $[r_i]_{2t}$ and

$$V_{n-t,n} \cdot \begin{pmatrix} [r_1]_t \\ [r_2]_t \\ \vdots \\ [r_n]_t \end{pmatrix} \qquad V_{n-t,n} \cdot \begin{pmatrix} [r_1]_{2t} \\ [r_2]_{2t} \\ \vdots \\ [r_n]_{2t} \end{pmatrix}, \qquad \text{e.g.,} \quad V_{n-t,n} = \begin{pmatrix} 1 & 2 & 3 & \dots & n \\ 1^2 & 2^2 & 3^2 & \dots & n^2 \\ & & & & \\ 1^{n-t} & 2^{n-t} & 3^{n-t} & \dots & n^{n-t} \end{pmatrix}$$

- Extract $n-t$ pairs out of n pairs
- $V_{n-t,n}$ is super-invertible (any $n-t$ by $n-t$ submatrix is invertible)

Batch Randomness Generation

How do we generate $[r]_t$, $[r]_{2t}$ efficiently?

- Each multiplication gate consumes one pair;
- Each pair should be generated with complexity not dependent on n

VanderMonde Randomness Extraction Damgard-Nielson'07

Each party generates a pair $[r_i]_t$, $[r_i]_{2t}$ and

$$V_{n-t,n} \cdot \begin{pmatrix} [r_1]_t \\ [r_2]_t \\ \vdots \\ [r_n]_t \end{pmatrix} \qquad V_{n-t,n} \cdot \begin{pmatrix} [r_1]_{2t} \\ [r_2]_{2t} \\ \vdots \\ [r_n]_{2t} \end{pmatrix}, \qquad \text{e.g.,} \quad V_{n-t,n} = \begin{pmatrix} 1 & 2 & 3 & \dots & n \\ 1^2 & 2^2 & 3^2 & \dots & n^2 \\ \\ 1^{n-t} & 2^{n-t} & 3^{n-t} & \dots & n^{n-t} \end{pmatrix}$$

- Extract $n-t$ pairs out of n pairs
- $V_{n-t,n}$ is super-invertible (any $n-t$ by $n-t$ submatrix is invertible)
- No matter which t parties are corrupted, the extracted masks are uniformly random.

Key Technical Barrier

For CRT secret sharing, how do we prove the security of these extracted masks?

$$V_{n-t,n} \cdot \begin{pmatrix} [r_1]_{2t} \\ [r_2]_{2t} \\ \vdots \\ [r_n]_{2t} \end{pmatrix} + \begin{pmatrix} D_1 \\ D_2 \\ \vdots \\ D_{n-t} \end{pmatrix} \approx V_{n-t,n} \cdot \begin{pmatrix} [r_1]_{2t} \\ [r_2]_{2t} \\ \vdots \\ [r_n]_{2t} \end{pmatrix} + \begin{pmatrix} D'_1 \\ D'_2 \\ \vdots \\ D'_{n-t} \end{pmatrix}$$

These are distributions over integers! Arguing statistical distance for distributions over integers is not easy.

High-dimensional Smudging Lemma

$$V_{n-t,n} \cdot \begin{pmatrix} [r_1]_{2t} \\ [r_2]_{2t} \\ \vdots \\ [r_n]_{2t} \end{pmatrix} + \begin{pmatrix} D_1 \\ D_2 \\ \vdots \\ D_{n-t} \end{pmatrix} \quad \text{and} \quad V_{n-t,n} \cdot \begin{pmatrix} [r_1]_{2t} \\ [r_2]_{2t} \\ \vdots \\ [r_n]_{2t} \end{pmatrix} + \begin{pmatrix} D'_1 \\ D'_2 \\ \vdots \\ D'_{n-t} \end{pmatrix}$$

for

$$V_{n-t,n} = \begin{pmatrix} 1 & 2 & 3 & \dots & n \\ 1^2 & 2^2 & 3^2 & \dots & n^2 \\ \\ 1^{n-t} & 2^{n-t} & 3^{n-t} & \dots & n^{n-t} \end{pmatrix}$$

are close as long as $D_i - D'_i$ are divisible by

$$\prod_{1 \leqslant i < j \leqslant n} (j - i)$$

High-dimensional Smudging Lemma

$$V_{n-t,n} \cdot \begin{pmatrix} [r_1]_{2t} \\ [r_2]_{2t} \\ \vdots \\ [r_n]_{2t} \end{pmatrix} + \begin{pmatrix} D_1 \\ D_2 \\ \vdots \\ D_{n-t} \end{pmatrix} \quad \text{and} \quad V_{n-t,n} \cdot \begin{pmatrix} [r_1]_{2t} \\ [r_2]_{2t} \\ \vdots \\ [r_n]_{2t} \end{pmatrix} + \begin{pmatrix} D'_1 \\ D'_2 \\ \vdots \\ D'_{n-t} \end{pmatrix}$$

for

$$V_{n-t,n} = \begin{pmatrix} 1 & 2 & 3 & \dots & n \\ 1^2 & 2^2 & 3^2 & \dots & n^2 \\ & & & & \\ 1^{n-t} & 2^{n-t} & 3^{n-t} & \dots & n^{n-t} \end{pmatrix}$$

are close as long as $D_i - D'_i$ are divisible by

$$\prod_{1 \leqslant i < j \leqslant n} (j - i)$$

- $\prod_{1 \leqslant i < j \leqslant n} (j - i)$ is a n^2-bit integer. To get rate-1, it means $\log |F|$ has to be $O(n^2)$.

High-dimensional Smudging Lemma

$$V_{n-t,n} \cdot \begin{pmatrix} [r_1]_{2t} \\ [r_2]_{2t} \\ \vdots \\ [r_n]_{2t} \end{pmatrix} + \begin{pmatrix} D_1 \\ D_2 \\ \vdots \\ D_{n-t} \end{pmatrix} \quad \text{and} \quad V_{n-t,n} \cdot \begin{pmatrix} [r_1]_{2t} \\ [r_2]_{2t} \\ \vdots \\ [r_n]_{2t} \end{pmatrix} + \begin{pmatrix} D'_1 \\ D'_2 \\ \vdots \\ D'_{n-t} \end{pmatrix}$$

for

$$V_{n-t,n} = \begin{pmatrix} 1 & 2 & 3 & \dots & n \\ 1^2 & 2^2 & 3^2 & \dots & n^2 \\ \\ 1^{n-t} & 2^{n-t} & 3^{n-t} & \dots & n^{n-t} \end{pmatrix}$$

are close as long as $D_i - D'_i$ are divisible by

$$\prod_{1 \leqslant i < j \leqslant n} (j - i)$$

- $\prod_{1 \leqslant i < j \leqslant n} (j - i)$ is a n^2-bit integer. To get rate-1, it means $\log |F|$ has to be $O(n^2)$.
- Due to proof techniques

Summary

Scalable MPC for general circuit over large prime field F:

- $|C| \cdot \log |F|$-bit communication/computation complexity
- Based on CRT-secret sharing
- "unpacked" secret sharing to achieve non-amortized rate-$O(1)$
- high-dimensional smudging lemma: randomness extraction over integers
- require $\log F = \widetilde{O}(n^2)$ — Open problem: can we prove the security for $\log F = \widetilde{O}(n)$?

Thanks!

Questions?