Preliminaries
oooooo

Building Blocks
oooooo

Calculating LUT
oooooooo

Benchmarks
oo

o

Secure Computing of Table with FLUTE

카이스트 전산학부 한승우

2024년 9월 6일

Lookup Table

LUT (Lookup Table)

δ-to-σ lookup table T is a function $T: \{0,1\}^{\delta} \to \{0,1\}^{\sigma}$.

x_1	x_2	x_3	y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

Example 3-to-1 LUT T

Goal   Given secret shares of x_1, x_2, and x_3, we want to compute the secret shares of $T(x_1, x_2, x_3)$.

Question   What secret sharing scheme should we use?

Question   How can we optimize the online and setup cost? (Up to 2σ bits per LUT caculation in the online phase!)

[·] **Secret Sharing**

$$P_0,\ P_1:\ \text{parties}$$

[·] Secret Sharing

A bit $v \in \mathbb{Z}_2$ is said to be [·]-*shared* between P_0 and P_1 if P_i holds $[v]_i$ such that

$$v = [v]_0 \oplus [v]_1.$$

$[\cdot]$ **Secret Sharing**

The $[\cdot]$-sharing is *linear* in the sense that:

1. For $[\cdot]$-shared $u, v \in \mathbb{Z}_2$, $[\cdot]$-sharing of $u \oplus v$ can be locally computed by:

$$[u \oplus v]_i = [u]_i \oplus [v]_i \text{ for each } i \in \{0, 1\}.$$

2. For $[\cdot]$-shared $v \in \mathbb{Z}_2$ and public $b \in \mathbb{Z}_2$, $[\cdot]$-sharing of bv can be locally computed by:

$$[bv]_i = b[v]_i \text{ for each } i \in \{0, 1\}.$$

3. For $[\cdot]$-shared $v \in \mathbb{Z}_2$ and public $b \in \mathbb{Z}_2$, $[\cdot]$-sharing of $v \oplus b$ can be locally computed by:

$$[v \oplus b]_i = [v]_i \oplus ib \text{ for each } i \in \{0, 1\}.$$

In particular, $[\cdot]$-sharing of $\overline{v} = v \oplus 1$ can be locally computed.

Multiplication of $[\cdot]$-shared Values

Beaver's Multiplication Triple

Suppose P_0 and P_1 have $[\cdot]$-shares of $u, v \in \mathbb{Z}_2$. Suppose
additionally that they have $[\cdot]$-shares of a, b, c where $c = ab$.

1. P_i computes $[u \oplus a]_i$ and $[v \oplus b]_i$, and sends them to P_{1-i}.

2. P_0 and P_1 now know $d := u \oplus a$ and $e := v \oplus b$.

3. P_i computes $[z]_i = i \cdot de \oplus d[b]_i \oplus e[a]_i \oplus [c]_i$.

4. Then, $[z]_0 \oplus [z]_1 = z = uv$.

With the prepared *multiplication triple*, a $[\cdot]$-share of uv is
calculated with the cost of four bits in a single round.

$$z = uv = (d \oplus a)(e \oplus b)$$
$$= \underbrace{de}_{\text{public}} \oplus db \oplus ea \oplus \underbrace{ab}_{=c}$$

⟨·⟩ Secret Sharing

⟨·⟩ Secret Sharing

A bit $v \in \mathbb{Z}_2$ is said to be ⟨·⟩-*shared* if:

1. A value $\lambda_v \in \mathbb{Z}_2$ is [·]-shared between P_0 and P_1.
   - $[\lambda_v]_i$'s are locally set arbitrarily.
2. A value $m_v \in \mathbb{Z}_2$ is known to both parties.
3. $v = m_v \oplus \lambda_v$

The ⟨·⟩-share of v is denoted $\langle v \rangle_i = (m_v, [\lambda_v]_i)$.

$\langle \cdot \rangle$ Secret Sharing

The $\langle \cdot \rangle$-sharing is *linear* in the sense that:

1. For $\langle \cdot \rangle$-shared $u, v \in \mathbb{Z}_2$, $\langle \cdot \rangle$-sharing of $u \oplus v$ can be locally computed by:

$$m_{u \oplus v} = m_u \oplus m_v \text{ and } [\lambda_{u \oplus v}]_i = [\lambda_u]_i \oplus [\lambda_v]_i$$

2. For $\langle \cdot \rangle$-shared $v \in \mathbb{Z}_2$ and public $b \in \mathbb{Z}_2$, $\langle \cdot \rangle$-sharing of bv can be locally computed by:

$$m_{bv} = b m_v \text{ and } [\lambda_{bv}]_i = b[\lambda_v]_i.$$

3. For $\langle \cdot \rangle$-shared $v \in \mathbb{Z}_2$ and public $b \in \mathbb{Z}_2$, $\langle \cdot \rangle$-sharing of $v \oplus b$ can be locally computed by:

$$m_{v \oplus b} = m_v \oplus b.$$

In particular, $\langle \cdot \rangle$-sharing of $\overline{v} = v \oplus 1$ can be locally computed.

Preliminaries
000000

Building Blocks
●00000

Calculating LUT
00000000

Benchmarks
00

○

Boolean Representation of Lookup Table

LUT Revisited

$$(\overline{x_1} \wedge \overline{x_2} \wedge \overline{x_3})$$
$$\vee(\overline{x_1} \wedge x_2 \wedge x_3)$$
$$\vee(x_1 \wedge \overline{x_2} \wedge x_3)$$

DNF representation of T

x_1	x_2	x_3	y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

Example 3-to-1 LUT T

$$(\overline{x_1} \wedge \overline{x_2} \wedge \overline{x_3})$$
$$\oplus(\overline{x_1} \wedge x_2 \wedge x_3)$$
$$\oplus(x_1 \wedge \overline{x_2} \wedge x_3)$$

An equivalent circuit.

$$y = \begin{pmatrix} \overline{x_1} \\ \overline{x_1} \\ x_1 \end{pmatrix} \odot \begin{pmatrix} \overline{x_2} \\ x_2 \\ \overline{x_2} \end{pmatrix} \odot \begin{pmatrix} \overline{x_3} \\ x_3 \\ x_3 \end{pmatrix}$$

"Multi-Fan-In Inner Product"

Multiplication of $\langle \cdot \rangle$-shared Values

$$z = uv = (\mathsf{m}_u \oplus \lambda_u)(\mathsf{m}_v \oplus \lambda_v)$$
$$= \underbrace{\mathsf{m}_u \mathsf{m}_v}_{\text{public}} \oplus \mathsf{m}_u \lambda_v \oplus \mathsf{m}_v \lambda_u \oplus \underbrace{\lambda_u \lambda_v}_{=\lambda_{uv}}$$

Multiplication of $\langle \cdot \rangle$-shared Values

Input $\langle \cdot \rangle$-shares of $u, v \in \mathbb{Z}_2$ and $[\cdot]$-shares of $\lambda_{uv} := \lambda_u \lambda_v$.

Output $\langle \cdot \rangle$-shares of uv.

1. P_i computes $[z]_i = i \cdot \mathsf{m}_u \mathsf{m}_v \oplus \mathsf{m}_u [\lambda_v]_i \oplus \mathsf{m}_v [\lambda_u]_i \oplus [\lambda_{uv}]_i$.

2. P_i randomly chooses $[\lambda_z]_i \in \mathbb{Z}_2$.

3. P_i computes $[\mathsf{m}_z]_i = [z]_i \oplus [\lambda_z]_i$ and sends it to P_{i-1}.

4. Then, P_0 and P_1 have $\langle \cdot \rangle$-shares of $z = uv$.

With the prepared multiplication triple, a $\langle \cdot \rangle$-share of uv is calculated with the cost of <u>two bits in a single round</u>.

Preliminaries
000000

Building Blocks
000000

Calculating LUT
00000000

Benchmarks
00

0

Building Blocks: Inner Product

Inner Product of $\langle \cdot \rangle$-shared Values

Inner Product of $\langle \cdot \rangle$-shared Values

Input $\langle \cdot \rangle$-shares of $u^1, \cdots, u^N, v^1 \cdots, v^N \in \mathbb{Z}_2$ and $[\cdot]$-shares of $\lambda_{u^1} \lambda_{v^1}, \cdots, \lambda_{u^N} \lambda_{v^N}$.

Output $\langle \cdot \rangle$-shares of $z = \bigoplus_{k=1}^{N} u^k v^k$.

1. P_i computes $[u^k v^k]_i$ for $k \in [N]$ as before.

2. P_i randomly chooses $[\lambda_z]_i \in \mathbb{Z}_2$.

3. P_i computes $[m_z]_i = [\lambda_z]_i \oplus \bigoplus_{k=1}^{N} [u^k v^k]_i$ and sends it to P_{1-i}.

4. Then, P_0 and P_1 have $\langle \cdot \rangle$-shares of $z = \bigoplus_{k=1}^{N} u^k v^k$.

With the prepared N multiplication triples, a $\langle \cdot \rangle$-share of $\bigoplus_{k=1}^{N} u^k v^k$ is calculated with the cost of <u>two bits in a single round</u>.

Multi-Fan-In Product of $\langle \cdot \rangle$-shared Values

Multi-Fan-In Product of $\langle \cdot \rangle$-shared Values

Input $\langle \cdot \rangle$-shares of $u_1, \cdots, u_M \in \mathbb{Z}_2$ and $\boxed{???}$.

Output $\langle \cdot \rangle$-shares of $\bigwedge_{j=1}^{M} u_j$.

Let $\mathcal{Q}$ be a set of $\langle \cdot \rangle$-shared values (wires). We use the following notation:
$$\mathsf{m}_{\mathcal{Q}} \triangleq \bigwedge_{u \in \mathcal{Q}} \mathsf{m}_u \quad \text{and} \quad \lambda_{\mathcal{Q}} \triangleq \bigwedge_{u \in \mathcal{Q}} \lambda_u.$$

Observation

$$\bigwedge_{j=1}^{M} u_j = \bigwedge_{j=1}^{M} (\mathsf{m}_{u_j} \oplus \lambda_{u_j}) = \bigoplus_{\mathcal{Q} \subseteq \mathcal{I}} (\mathsf{m}_{\mathcal{Q}} \cdot \lambda_{\mathcal{I} \setminus \mathcal{Q}})$$

where $\mathcal{I} := \{ u_1, u_2, \cdots, u_M \}$.

Multi-Fan-In Product of $\langle\cdot\rangle$-shared Values

Input $\langle\cdot\rangle$-shares of $u_1, \cdots, u_M \in \mathbb{Z}_2$ and $[\cdot]$-shares of $\lambda_{\mathcal{Q}}$ for each $\mathcal{Q} \subseteq \mathcal{I}$ with $|\mathcal{Q}| > 1$ where $\mathcal{I} = \{u_1, \cdots, u_M\}$.

Output $\langle\cdot\rangle$-shares of $\bigwedge_{j=1}^{M} u_j$.

1. P_i computes
$$[z]_i = i \cdot m_{\mathcal{I}} \oplus \bigoplus_{\mathcal{Q} \subsetneq \mathcal{I}} (m_{\mathcal{Q}} \cdot [\lambda_{\mathcal{I} \setminus \mathcal{Q}}]_i).$$

2. P_i randomly chooses $[\lambda_z]_i \in \mathbb{Z}_2$.

3. P_i computes $[m_z]_i = [z]_i \oplus [\lambda_z]_i$ and sends it to P_{1-i}.

4. Then, P_0 and P_1 have $\langle\cdot\rangle$-shares of $\bigwedge_{j=1}^{M} u_j$.

With the prepared $2^M - M - 1$ multiplication triples, a $\langle\cdot\rangle$-share of $\bigwedge_{j=1}^{M} u_j$ is calculated with the cost of <u>two bits in a single round</u>.

Multi-Fan-In Inner Product of $\langle\cdot\rangle$-shared Values

Input $\langle\cdot\rangle$-shares of $\mathbf{u}^1, \cdots, \mathbf{u}^M \in (\mathbb{Z}_2)^N$ and $[\cdot]$-shares of $\lambda_{\mathcal{Q}}$ for each $\mathcal{Q} \subseteq \mathcal{I}_k$ with $|\mathcal{Q}| > 1$ where $\mathcal{I}_k = \{\mathbf{u}_k^1, \cdots, \mathbf{u}_k^M\}$ for $k \in [N]$.

Output $\langle\cdot\rangle$-shares of $\mathbf{u}^1 \odot \cdots \odot \mathbf{u}^N = \bigoplus_{k=1}^N \bigwedge_{j=1}^M \mathbf{u}_k^j$.

1. P_i computes

$$[z]_i = i \cdot \bigoplus_{k=1}^N \mathsf{m}_{\mathcal{I}_k} \oplus \bigoplus_{k=1}^N \bigoplus_{\mathcal{Q} \subsetneq \mathcal{I}_k} (\mathsf{m}_{\mathcal{Q}} \cdot [\lambda_{\mathcal{I} \setminus \mathcal{Q}}]_i).$$

2. P_i randomly chooses $[\lambda_z]_i \in \mathbb{Z}_2$.

3. P_i computes $[\mathsf{m}_z]_i = [z]_i \oplus [\lambda_z]_i$ and sends it to P_{1-i}.

4. Then, P_0 and P_1 have $\langle\cdot\rangle$-shares of $\mathbf{u}^1 \odot \cdots \odot \mathbf{u}^N$.

With the prepared $N(2^M - M - 1)$ multiplication triples, a $\langle\cdot\rangle$-share of $\bigodot_{j=1}^N \mathbf{u}^j$ is calculated with the cost of <u>two bits in a single round</u>.

Input $\langle\cdot\rangle$-shares of $\mathbf{u}^1, \cdots, \mathbf{u}^M \in (\mathbb{Z}_2)^N$ and $[\cdot]$-shares of $\lambda_{\mathcal{Q}}$ for each $\mathcal{Q} \subseteq \mathcal{I}_k$ with $|\mathcal{Q}| > 1$ where $\mathcal{I}_k = \{\mathbf{u}_k^1, \cdots, \mathbf{u}_k^M\}$ for $k \in [N]$.

Output $\langle\cdot\rangle$-shares of $\mathbf{u}^1 \odot \cdots \odot \mathbf{u}^N = \bigoplus_{k=1}^N \bigwedge_{j=1}^M \mathbf{u}_k^j$.

With the prepared $\underline{N(2^M - M - 1)}$ multiplication triples, a $\langle\cdot\rangle$-share of $\bigodot_{j=1}^N \mathbf{u}^j$ is calculated with the cost of two bits in a single round.

$$y = \begin{pmatrix} \overline{x_1} \\ \overline{x_1} \\ x_1 \end{pmatrix} \odot \begin{pmatrix} \overline{x_2} \\ x_2 \\ \overline{x_2} \end{pmatrix} \odot \begin{pmatrix} \overline{x_3} \\ x_3 \\ x_3 \end{pmatrix}$$

For δ-to-1 LUT,
$N \leq 2^{\delta-1}$ and $M = \delta$??
$2^{\delta-1}(2^\delta - \delta - 1)$ triples??

We now exploit the fact that elements of $\mathbf{u}^j$ are closely related.

λ_v is unchanged when calculating shares of $\overline{v}$!

Some Definitions

Fix a δ-to-σ LUT T.

x^1	x^2	x^3	y^1	y^2
0	0	0	1	0
0	0	1	0	0
0	1	0	0	0
0	1	1	1	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	0
1	1	1	0	1
$\mathbf{e}^1$	$\mathbf{e}^2$	$\mathbf{e}^3$	$\mathbf{y}^1$	$\mathbf{y}^2$

For instance, in this example,

$$
\mathbf{e}^2 = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}
\quad \text{and} \quad
\mathbf{y}^1 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}.
$$

Expressing LUT with Multi-Fan-In Inner Product

Now, in the previous example, note that:

<div>

x^1	x^2	x^3	y^1
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

</div>

$$\mathbf{z}^1 = \begin{pmatrix} \overline{x^1} \\ \overline{x^1} \\ x^1 \end{pmatrix} \odot \begin{pmatrix} \overline{x^2} \\ x^2 \\ \overline{x^2} \end{pmatrix} \odot \begin{pmatrix} \overline{x^3} \\ x^3 \\ x^3 \end{pmatrix}$$

$$= \begin{pmatrix} \overline{x^1} \\ \overline{x^1} \\ \overline{x^1} \\ \overline{x^1} \\ \vdots \end{pmatrix} \odot \begin{pmatrix} \overline{x^2} \\ \overline{x^2} \\ x^2 \\ x^2 \\ \vdots \end{pmatrix} \odot \begin{pmatrix} \overline{x^3} \\ x^3 \\ \overline{x^3} \\ x^3 \\ \vdots \end{pmatrix} \odot \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ \vdots \end{pmatrix}$$

$$= (\overline{x^1} \oplus \mathbf{e}^1) \odot \cdots \odot (\overline{x^3} \oplus \mathbf{e}^3) \odot \mathbf{y}^1$$

$\overline{x^j} \oplus \mathbf{e}^j$ is an element-wise addition.

Expressing LUT with Multi-Fan-In Inner Product

Fixing a δ-to-σ LUT T, and given $\langle\cdot\rangle$-shares of $x^1,\cdots,x^\delta$, let

$$\mathbf{u}^j := \overline{x^j} \oplus \mathbf{e}^j \in (\mathbb{Z}_2)^{2^\delta}$$

for $j \in [\delta]$ so that

$$\langle\mathbf{u}_k^j\rangle_i = \left(\overline{\mathsf{m}_{x^j}} \oplus \mathbf{e}_k^j, [\lambda_{x^j}]_i\right)$$

gives a $\langle\cdot\rangle$-sharing of $\mathbf{u}_k^j$ for $j \in [\delta]$, $k \in [2^\delta]$.

With this, we have

$$\lambda_{\mathbf{u}_{k_1}^{j_1}\cdots\mathbf{u}_{k_n}^{j_n}} = \lambda_{x^{j_1}\ldots x^{j_n}}!$$

Expressing LUT with Multi-Fan-In Inner Product

As previously shown, the w-th output $\mathbf{z}^w$ can be calculated by

$$
\begin{aligned}
\mathbf{z}^w &= (\overline{x^1} \oplus \mathbf{e}^1) \odot \cdots \odot (\overline{x^\delta} \oplus \mathbf{e}^\delta) \odot \mathbf{y}^w \\
&= \mathbf{u}^1 \odot \cdots \odot \mathbf{u}^\delta \odot \mathbf{y}^w \\
&= \bigoplus_{k=1}^{2^\delta} \left[\left(\bigwedge_{j=1}^{\delta} \mathbf{u}_k^j \right) \wedge \mathbf{y}_k^w \right] \\
&= \bigoplus_{k=1}^{2^\delta} \left[\left(\bigwedge_{j=1}^{\delta} \left(\mathrm{m}_{\mathbf{u}_k^j} \oplus \lambda_{\mathbf{u}_k^j} \right) \right) \wedge \mathbf{y}_k^w \right] \\
&= \bigoplus_{k=1}^{2^\delta} \bigoplus_{\mathcal{Q}_k \subseteq \mathcal{I}_k} \left(\mathrm{m}_{\mathcal{Q}_k} \cdot \mathbf{y}_k^w \cdot \lambda_{\mathcal{I}_k \setminus \mathcal{Q}_k} \right).
\end{aligned}
$$

Preliminaries	Building Blocks	Calculating LUT	Benchmarks	
000000	000000	00000●00	00	0

Calculating LUT

Calculating LUT

Calculating δ-to-σ LUT

Input A δ-to-σ LUT T, $\langle \cdot \rangle$-shares of $x^1, \cdots, x^\delta \in \mathbb{Z}_2$, and
       $[\cdot]$-shares of $\lambda_{\mathcal{Q}}$ for each $\mathcal{Q} \subseteq \mathcal{I}$ with $|\mathcal{Q}| > 1$ where
       $\mathcal{I} = \{x^1, \cdots, x^\delta\}$.

Output $\langle \cdot \rangle$-shares of $T(x^1, \cdots, x^\delta)$.

1. P_i computes its share of $\mathbf{u}_k^j = \overline{x^j} \oplus \mathbf{e}_k^j$ for $j \in [\delta]$ and $k \in [2^\delta]$.

2. P_i computes, for each $w \in [\sigma]$,
$$[\mathbf{z}_w]_i = i \cdot \bigoplus_{k=1}^{2^\delta} (m_{\mathcal{I}_k} \cdot \mathbf{y}_k^w) \oplus \bigoplus_{k=1}^{2^\delta} \bigoplus_{\mathcal{Q}_k \subsetneq \mathcal{I}_k} \left(m_{\mathcal{Q}_k} \cdot \mathbf{y}_k^w \cdot [\lambda_{\mathcal{I}_k \setminus \mathcal{Q}_k}]_i \right).$$

3. P_i randomly chooses $[\lambda_{\mathbf{z}_w}]_i \in \mathbb{Z}_2$ for $w \in [\sigma]$.

4. P_i computes $[m_{\mathbf{z}_w}]_i = [\mathbf{z}_w]_i \oplus [\lambda_{\mathbf{z}_w}]_i$ for $w \in [\sigma]$ and sends
   them to P_{1-i}.

5. Then, P_0 and P_1 have $\langle \cdot \rangle$-shares of $\mathbf{z} = T(x^1, \cdots, x^\delta)$.

Calculating LUT

Calculating δ-to-σ LUT

Input A δ-to-σ LUT T, $\langle \cdot \rangle$-shares of $x^1, \cdots, x^\delta \in \mathbb{Z}_2$, and
$[\cdot]$-shares of $\lambda_{\mathcal{Q}}$ for each $\mathcal{Q} \subseteq \mathcal{I}$ with $|\mathcal{Q}| > 1$ where
$\mathcal{I} = \{x^1, \cdots, x^\delta\}$.

Output $\langle \cdot \rangle$-shares of $\mathsf{T}(x^1, \cdots, x^\delta)$.

With the prepared $2^\delta - \delta - 1$ multiplication triples, a $\langle \cdot \rangle$-share of
$\mathsf{T}(x^1, \cdots, x^\delta)$ is calculated with the cost of
2σ bits in a single round.

Pros and Cons

Pros

- Any complex circuit can be represented by a *interconnection of small LUTs*.
- Evaluation does not depend on the internal logic.
- Setup does not depend on the number of outputs.

Cons

- Exponential setup time/communicaion.
- Exponential internal calculation.

Preliminaries
○○○○○○

Building Blocks
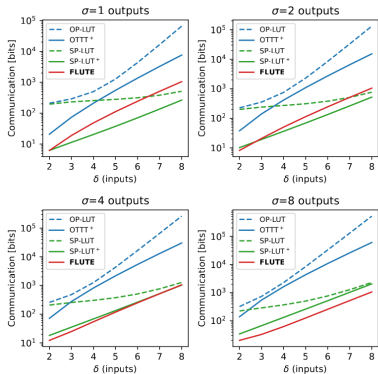○○○○○○

Calculating LUT
○○○○○○○○

Benchmarks
●○

○

Benchmarks

Analytic Comparison with Other Protocols

Figure 7: Total communication for different LUT sizes with $2 \leq \delta \leq 8$ inputs and $\sigma \in \{1, 2, 4, 8\}$ outputs.
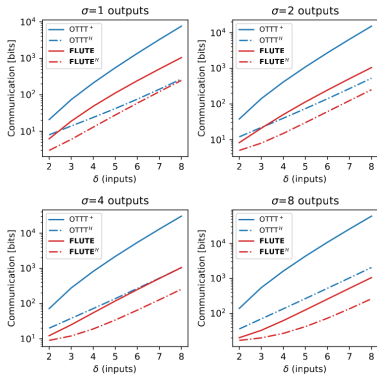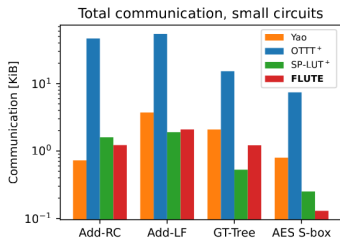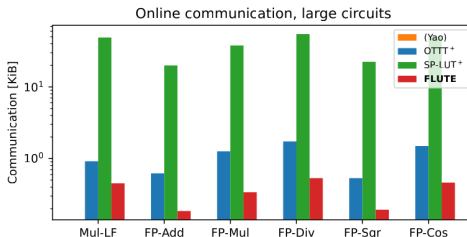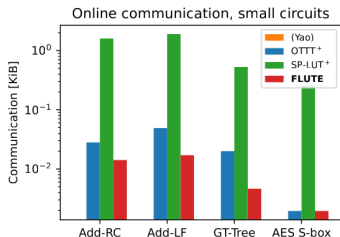
Figure 8: Total communication for different LUT sizes with $2 \leq \delta \leq 8$ inputs and $\sigma \in \{1, 2, 4, 8\}$ outputs using a helper server $S_{\mathcal{H}}$ (cf. §C).

Preliminaries
○○○○○○

Building Blocks
○○○○○○

Calculating LUT
○○○○○○○○○

Benchmarks
○●

○

Benchmarks

Online/Total Communication Compared with Other Protocols
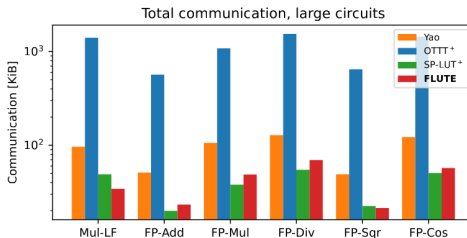
References

- Brüggemann, A., Hundt, R., Schneider, T., Suresh, A., & Yalame, H. (2023). Flute: Fast and secure lookup table evaluations. *2023 IEEE Symposium on Security and Privacy (SP)*.
  https://doi.org/10.1109/sp46215.2023.10179345