GCWise
Garbled Circuits with Sublinear Evaluator

Abida Haque
David Heath
Vladimir Kolesnikov
Steve Lu
Rafail Ostrovsky
Akash Shah

NC State
Georgia Tech ⇒ UIUC
Georgia Tech
Stealth Software Technologies, Inc.
UCLA
UCLA
\( \hat{C} \) to \( \hat{C} \) through the process \( C \)

- **Generator**
- **Evaluator**
\[ \hat{C} \]

\[ C_0 \quad C_1 \quad \ldots \quad C_{n-1} \]

Generator

Evaluator
Goal: Sublinear in $n$

$\hat{C}$

Generator

$C_0$ $C_1$ ... $C_{n-1}$

Evaluator
Goal: Sublinear in $n$

$\hat{C}$

Generator

$C_0 \quad C_1 \quad \ldots \quad C_{n-1}$

Evaluator

$\hat{C}$
For functions with conditionals, achieve **sublinear*** communication and **sublinear*** computation for one party

* sublinear in the function description
For functions with conditionals, achieve **compact 2PC**
Compact 2PC

FHE allows compact 2PC, but requires expensive primitives.

Stacked Garbling (Heath, Kolesnikov Crypto 2020) does not achieve compactness.
Compact 2PC

FHE allows compact 2PC, but requires expensive primitives.

Stacked Garbling (Heath, Kolesnikov Crypto 2020) does not achieve compactness.

Our starting point.
\[ f(i, x) = \begin{cases} 
C_0(x) & \text{if } i = 0 \\
C_1(x) & \text{if } i = 1 \\
\vdots \\
C_{n-1}(x) & \text{if } i = n - 1 
\end{cases} \]

\( \tilde{O}(\sqrt{n}) \) communication

\( \tilde{O}(\sqrt{n}) \) evaluator computation
Stacked Garbling

Generator

\[ C_0 \quad C_1 \quad \ldots \quad C_{n-1} \]
Stacked Garbling

$S_0 \quad S_1 \quad S_{n-1}$

$C_0 \quad C_1 \quad \ldots \quad C_{n-1}$

Generator
Stacked Garbling

$S_0, S_1, S_{n-1}$

$C_0, C_1, \ldots, C_{n-1}$

$\hat{C}_0, \hat{C}_1, \hat{C}_{n-1}$

Generator
Stacked Garbling

Generator

\[ C_0 \quad C_1 \quad \ldots \quad C_{n-1} \]

\[ \hat{C}_0 \quad \hat{C}_1 \quad \hat{C}_{n-1} \]
Stacked Garbling

Size independent of $n$
Stacked Garbling

... where Evaluator knows the active branch
Stacked Garbling

... where Evaluator knows the active branch

\[ S_0 \quad \bigoplus \quad C_1 \quad \cdots \quad \bigoplus \quad C_{n-1} \]

Evaluator
Stacked Garbling

\[ \hat{C}_0 \oplus \hat{C}_i \]

... where Evaluator knows the active branch
Stacked Garbling

... where Evaluator knows the active branch

\[ \hat{C}_0 \oplus \hat{C}_1 \oplus \hat{C}_{n-1} \]
Stacked Garbling achieves sublinear communication…

But not sublinear Evaluator computation
\( \tilde{O}(\sqrt{n}) \)  

Bucket Content

Generator

Buckets
\[ \tilde{O}(\sqrt{n}) \] Bucket Content

\[
\begin{array}{c}
\boxed{C_{17}} \\
\ldots \\
\ldots \\
\end{array}
\]

Generator

Buckets
\(\tilde{O}(\sqrt{n})\) Bucket Content

\[
C_{17} \quad C_{42}
\]

Generator

\(n\)
\(\tilde{O}(\sqrt{n})\) Bucket Content

\[
\begin{array}{ccc}
C_{17} & C_{42} & C_5 \\
\end{array}
\]
\[ \tilde{O}(\sqrt{n}) \] Buckets

\[ C_{17}, C_{42}, C_5, C_{42} \]

Generator

Bucket Content
$\tilde{O}(\sqrt{n})$

Bucket Content

$C_{17} \quad C_{42} \quad C_5 \quad C_{42} \ldots$

Generator
\tilde{O}(\sqrt{n}) \quad \text{Bucket Content}

\[
\begin{array}{cccc}
C_{17} & C_{42} & C_5 & C_{42} \\
C_3 & C_0 & C_{51} & C_{14} \\
C_{21} & C_{25} & C_6 & C_0 \\
\vdots & \vdots & \vdots & \vdots \\
\end{array}
\]
Generator

\[
\begin{array}{cccc}
C_{17} & C_{42} & C_5 & C_{42} \\
C_3 & C_0 & C_{51} & C_{14} \\
C_{21} & C_{25} & C_6 & C_0 \\
\end{array}
\]

...
\(\tilde{O}(\sqrt{n})\)
Universal Circuit

$C_0 \rightarrow C_?$
$O(\log |C|)$ overhead
$O(1)$ overhead in special cases
Evaluator
\( \tilde{O}(\sqrt{n}) \)

Evaluator

\[ C \_ C \_ C \_ C \_ \ldots \oplus \hat{C} \]
$C_3$  $C_0$  $C?$  $C_{14}$  ...  $\hat{C}$
Evaluator
Evaluator
Evaluator
Evaluator

\[ C_3 \quad C_0 \quad C_? \quad C_{14} \quad \ldots \quad \bigoplus \hat{C} \]

\[ \hat{C}_3 \quad \hat{C}_0 \quad \hat{C}_? \quad \hat{C}_{14} \]

\[ \tilde{O}(\sqrt{n}) \]
Surprising technical challenge:

Compactly identify and reveal to the Evaluator the active branch/siblings
Key Insight:

Pseudorandomly select the content of one bucket
Key Insight:

Pseudorandomly select the content of one bucket

Choose the content of the other buckets based off this first bucket
GCWise

- Organize branches into $\tilde{O}(\sqrt{n})$ buckets
- For each bucket, stack the branches
- Evaluator considers only the single active bucket
Garbled PIR
Read 1-out-of-$n$ database items at cost compact in $n$
$C_0(\cdot) \mapsto x_0$
\[ C_0(\cdot) \mapsto x_0 \]
\[ C_1(\cdot) \mapsto x_1 \]
\[ C_2(\cdot) \mapsto x_2 \]
\[ C_3(\cdot) \mapsto x_3 \]
\[ C_4(\cdot) \mapsto x_4 \]
\[ \vdots \]
\[ C_{n-1}(\cdot) \mapsto x_{n-1} \]
GCWise

- Compact 2PC for functions with conditionals
- Garbled PIR