Secure Computation on Floating Point Numbers

Mehrdad Aliasgari, Marina Blanton, Yihua Zhang, and Aaron Steele
University of Notre Dame

February 27, 2013

Outline

Outline

Outline

SMC

A number of parties ($n > 2$) wish to jointly and **securely** compute a known function (F) on their private inputs.

- Privacy-Preserving Computation
- Secure Outsourcing / Cloud Computation
- Secure Collaborative Computation

SMC-Cont.

Recent progress has made it fast.

- Generally, any computable function can be evaluated securely (e.g., as a Boolean or arithmetic circuit)
- Optimization of existing techniques

SMC-Cont.

Recent progress has made it fast.

- Generally, any computable function can be evaluated securely (e.g., as a Boolean or arithmetic circuit)
- Optimization of existing techniques
- Mainly integer domain
- Little attempt on real numbers

SMC-Cont.

Recent progress has made it fast.

- Generally, any computable function can be evaluated securely (e.g., as a Boolean or arithmetic circuit)
- Optimization of existing techniques
- Mainly integer domain
- Little attempt on real numbers
- NO Floating point support in SMC

Outline

Secret Sharing

Linear Secret Sharing scheme (such as Shamir secret sharing scheme [1])

- $P_1 \cdots P_n$ parties engage in a (n, t)-secret sharing scheme $(t < n/2)$
- $[x]$
- Linear combination of secrets can be computed locally
- Multiplication of two secrets requires one round of an interactive operation
- Performance Metric: $\#$ of interactive operations along with $\#$ of sequential interactions (rounds)

[1] A. Shamir. How to share a secret. Communications of the ACM, 1979

Outline

- $[b] \leftarrow LT([x],[y],\ell)$ Catrina and de Hoogh's takes 4 rounds and $4\ell - 2$ interactive operations.
- $[y] \leftarrow Trunc([x],\ell,m)$ 4 rounds and $4m+1$ interactions
- $[x_{m-1}] \cdots [x_0] \leftarrow BitDec([x],\ell,m)$ $\log m$ rounds and $m \log(m)$ interactions

Outline

FL Representation

$$u = (1 - z)(1 - 2s)v2^p$$

- Normalized Value $v \in [2^{\ell-1}, 2^{\ell})$
- Power $p \in (-2^{k-1}, 2^{k-1})$
- Sign indicator $s = \{0, 1\}$
- Zero indicator $z = \{0, 1\}$
  - $u = 0 \Leftrightarrow z = 1, v = p = 0$

FL Representation

$u = (1 - z)(1 - 2s)v2^p$

- Normalized Value $v \in [2^{\ell-1}, 2^{\ell})$
- Power $p \in (-2^{k-1}, 2^{k-1})$
- Sign indicator $s = \{0, 1\}$
- Zero indicator $z = \{0, 1\}$
  - $u = 0 \Leftrightarrow z = 1, v = p = 0$

Error Detection:

- *Invalid operation*
- *Division by zero*
- *Overflow and Underflow*

FL Representation

$u = (1 - z)(1 - 2s)v2^p$

- Normalized Value $v \in [2^{\ell-1}, 2^{\ell})$
- Power $p \in (-2^{k-1}, 2^{k-1})$
- Sign indicator $s = \{0, 1\}$
- Zero indicator $z = \{0, 1\}$
  - $u = 0 \Leftrightarrow z = 1, v = p = 0$

Error Detection:

- *Invalid operation*
- *Division by zero*
- *Overflow and Underflow*
- *Inexact*

Outline

- $[y] \leftarrow \mathrm{Trunc}([a], \ell, [m])$
  - $O(\ell)$ invocations and $O(\log\log\ell)$ rounds
- $[a_0], \ldots, [a_{\ell-1}] \leftarrow \mathrm{B2U}([a], \ell)$
  - $O(\ell)$ invocations and $O(\log\log\ell)$ rounds
- $[2^a] \leftarrow \mathrm{Pow2}([a], \ell)$
  - $O((\log\ell)(\log\log\ell))$ invocations and $O(\log\log\ell)$ rounds

Outline

Basic FL-1

- $\langle [v], [p], [z], [s] \rangle \leftarrow \mathsf{FLMul}(\langle [v_1], [p_1], [z_1], [s_1] \rangle, \langle [v_2], [p_2], [z_2], [s_2] \rangle)$
  - $O(\ell)$ invocations and $O(1)$ rounds

Basic FL-1

- $\langle [v], [p], [z], [s] \rangle \leftarrow \mathsf{FLMul}(\langle [v_1], [p_1], [z_1], [s_1] \rangle, \langle [v_2], [p_2], [z_2], [s_2] \rangle)$
  - $O(\ell)$ invocations and $O(1)$ rounds
- $\langle [v], [p], [z], [s] \rangle \leftarrow \mathsf{FLDiv}(\langle [v_1], [p_1], [z_1], [s_1] \rangle, \langle [v_2], [p_2], [z_2], [s_2] \rangle)$
  - $O(\ell \log \ell)$ invocations and $O(\log \ell)$ rounds

Basic FL-1

- $\langle [v], [p], [z], [s] \rangle \leftarrow \mathsf{FLMul}(\langle [v_1], [p_1], [z_1], [s_1] \rangle, \langle [v_2], [p_2], [z_2], [s_2] \rangle)$
  - $O(\ell)$ invocations and $O(1)$ rounds
- $\langle [v], [p], [z], [s] \rangle \leftarrow \mathsf{FLDiv}(\langle [v_1], [p_1], [z_1], [s_1] \rangle, \langle [v_2], [p_2], [z_2], [s_2] \rangle)$
  - $O(\ell \log \ell)$ invocations and $O(\log \ell)$ rounds
- $\langle [v], [p], [z], [s] \rangle \leftarrow \mathsf{FLAdd}(\langle [v_1], [p_1], [z_1], [s_1] \rangle, \langle [v_2], [p_2], [z_2], [s_2] \rangle)$
  - $O(\ell \log \ell + k)$ invocations and $O(\log \ell)$ rounds

Basic FL-2

- $[b] \leftarrow \mathsf{FLLT}(\langle [v_1], [p_1], [z_1], [s_1] \rangle, \langle [v_2], [p_2], [z_2], [s_2] \rangle)$
  - $O(\ell + k)$ invocations and $O(1)$ rounds
- $\langle [v], [p], [z], [s] \rangle \leftarrow \mathsf{FLRound}(\langle [v_1], [p_1], [z_1], [s_1] \rangle, \mathrm{mode})$
  - mode $= 0 \rightarrow$ floor and mode $= 1 \rightarrow$ ceiling
  - $O(\ell + k)$ invocations and $O(\log \log \ell)$ rounds

FLRound

$\langle [v], [p], [z], [s] \rangle \leftarrow \mathsf{FLRound}(\langle [v_1], [p_1], [z_1], [s_1] \rangle, \mathsf{mode})$

- $[a] \leftarrow \mathsf{LTZ}([p_1], k);$
- $[b] \leftarrow \mathsf{LT}([p_1], -\ell + 1, k)$
- $\langle [v_2], [2^{-p_1}] \rangle \leftarrow \mathsf{Mod2m}([v_1], \ell, -[a](1 - [b])[p_1]);$
- $[c] \leftarrow \mathsf{EQZ}([v_2], \ell);$
- $[v] \leftarrow [v_1] - [v_2] + (1 - [c])[2^{-p_1}](\mathsf{XOR}(\mathsf{mode}, [s_1]));$
- $[d] \leftarrow \mathsf{EQ}([v], 2^{\ell}, \ell + 1);$
- $[v] \leftarrow [d]2^{\ell-1} + (1 - [d])[v];$
- $[v] \leftarrow [a]((1 - [b])[v] + [b](\mathsf{mode} - [s_1])) + (1 - [a])[v_1];$
- $[s] \leftarrow (1 - [b]\mathsf{mode})[s_1];$
- $[z] \leftarrow \mathsf{OR}(\mathsf{EQZ}([v], \ell), [z_1]);$
- $[v] \leftarrow [v](1 - [z]);$
- $[p] \leftarrow ([p_1] + [d][a](1 - [b]))(1 - [z]);$

Outline

- $\langle[v],[p],[z],[s]\rangle \leftarrow \mathsf{FLSqrt}(\langle[v_1],[p_1],[z_1],[s_1]\rangle)$
- $\langle[v],[p],[z],[s]\rangle \leftarrow \mathsf{FLExp2}(\langle[v_1],[p_1],[z_1],[s_1]\rangle)$
- $\langle[v],[p],[z],[s]\rangle \leftarrow \mathsf{FLLog2}(\langle[v_1],[p_1],[z_1],[s_1]\rangle)$

Outline

- Integer: γ bits
- Fixed point: $u = \overline{u}2^{-f}$
  - $\overline{u}$: signed γ-bit integer
- Floating Point: $u = (1-2s)(1-z)v2^p$
  - v: normalized ℓ-bit value and
  - p: signed k-bit exponent
  - $k > \max(\lceil \log(\ell + f) \rceil, \lceil \log(\gamma) \rceil)$

Conversion-cont.

- $\langle [v], [p], [z], [s] \rangle \leftarrow \mathsf{Int2FL}([a], \gamma, \ell)$

Conversion-cont.

- $\langle [v], [p], [z], [s] \rangle \leftarrow \mathsf{Int2FL}([a], \gamma, \ell)$
- $[g] \leftarrow \mathsf{FL2Int}(\langle [v], [p], [z], [s] \rangle, \ell, k, \gamma)$

Conversion-cont.

- $\langle [v], [p], [z], [s] \rangle \leftarrow \mathsf{Int2FL}([a], \gamma, \ell)$
- $[g] \leftarrow \mathsf{FL2Int}(\langle [v], [p], [z], [s] \rangle, \ell, k, \gamma)$
- $\langle [v], [p], [z], [s] \rangle \leftarrow \mathsf{FP2FL}([g], \gamma, f, \ell, k)$
- $[g] \leftarrow \mathsf{FL2FP}(\langle [v], [p], [z], [s] \rangle, \ell, k, \gamma, f)$

Outline

Security

- Cannetti's composition theorem
    - R. Canetti. "Security and composition of multiparty cryptographic protocols." Journal of Cryptology, 2000.
- Secure in the malicious adversaries model

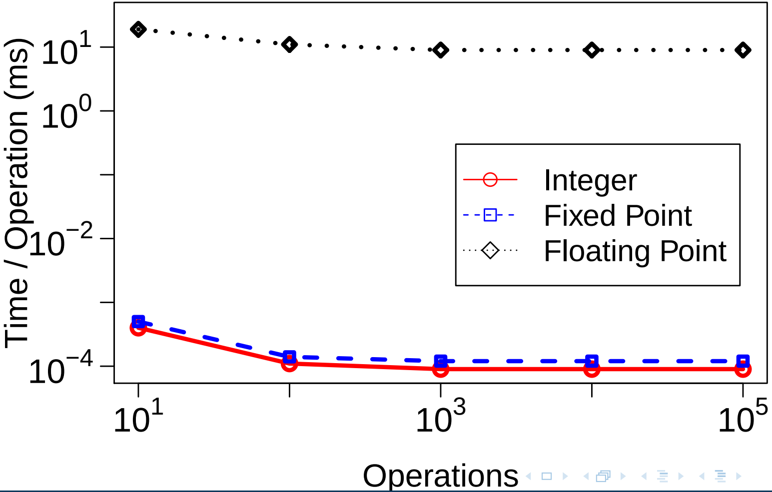
Experiments

- Integer: $\gamma = 64$
  - $|q| > 2\gamma + \kappa + 1 = 177$
- Fixed point: $\gamma = 64$ and $f = 32$ (precision : 2^{-32})
  - $|q| > \gamma + 3f + \kappa = 208$
- Floating point: $\ell = 32$ and $k = 9$ (precision : 2^{-256})
  - $|q| > 2\ell + \kappa + 1 = 113$

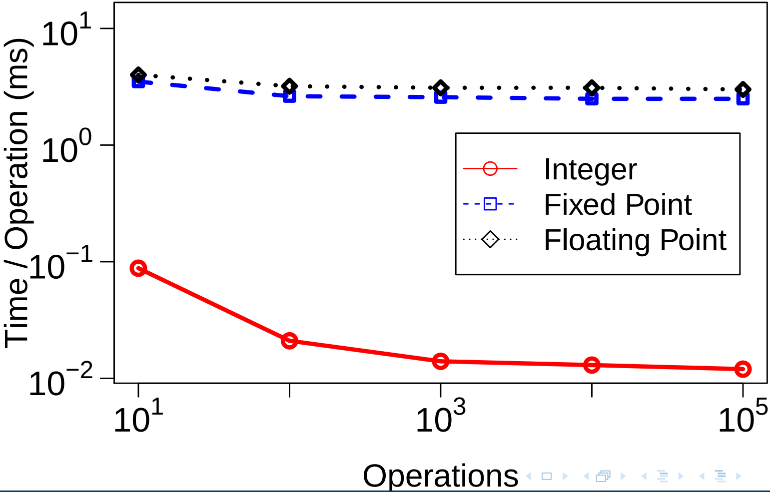
Experiments Cont.

- C/C++ using the GMP library
- (3, 1)-Shamir secret sharing
- Boost libraries for communication and OpenSSL for securing the communication
- 2.2 GHz Linux machines on a 1Gbps LAN

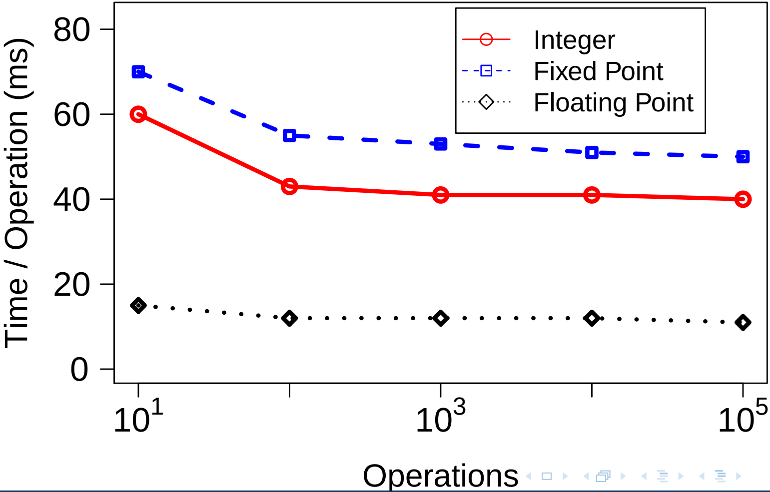
Addition


Time / Operation (ms)
10^1
10^0
10^{-2}
10^{-4}
Integer
Fixed Point
Floating Point
10^1
10^3
10^5
Operations

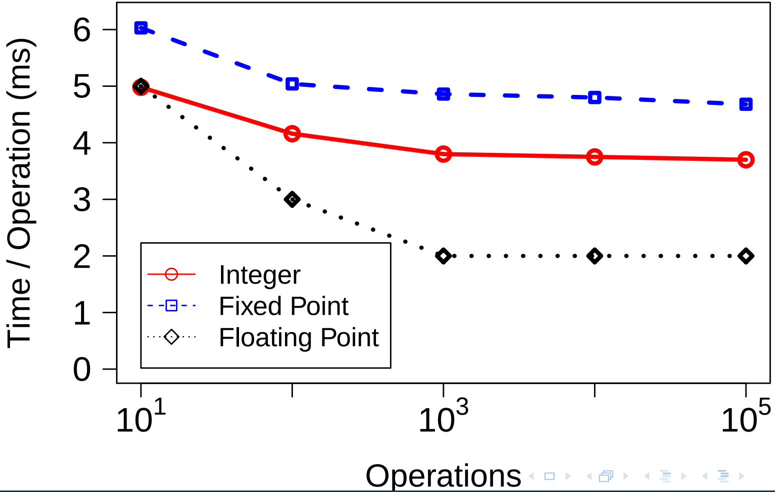
Multiplication



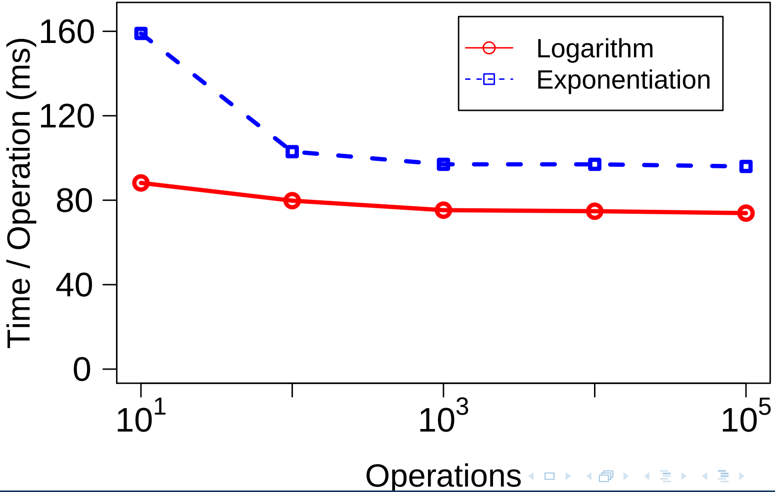
Division


Integer
Fixed Point
Floating Point
80
60
40
20
0
Time / Operation (ms)
10^1
10^3
10^5
Operations

Comparison


Time / Operation (ms)
6
5
4
3
2
1
0
Integer
Fixed Point
Floating Point
10^1
10^3
10^5
Operations

Exp & Log


Logarithm
Exponentiation
Time / Operation (ms)
160
120
80
40
0
10^1
10^3
10^5
Operations