Centrum Wiskunde & Informatica

Near-Linear Unconditionally-Secure MPC with a Dishonest Minority

Serge Fehr

CWI Amsterdam

www.cwi.nl/~fehr

Eli Ben-Sasson

Technion

Rafail Ostrovsky

UCLA

Multiparty Computation (MPC)

Goal:

Compute function f on private inputs $x_1,\dots,x_n$, so that

- all learn correct $f(x_1,\dots,x_n)$
- x_i's remain private

even if adversary corrupts t players.

Classical possibility results:

- computational security for $t < n/2$ [GMW87,CDG88]
- unconditional security for $t < n/2$ (assuming broadcast) [RB89,Bea89]
- perfect security for $t < n/3$ [CCD88,BGW88]

Beyond (im)possibility results: (communication) complexity

Amortized Communication Complexity

- Best known results (binary circuits):

Attack	Resilience	Security	Bits/multiplication [1)]	Ref
passive	$t < n/2$	perfect	$O(n \log n)$	[DamNie07]
active	$t < n/2$	computational	$O(n \log n)$	[DamNie07]
active	$t < n/2$	unconditional	$O(n^2 k)$	[BerHirt06]
active	$t < n/3$	perfect	$O(n \log n)$ [2)]	[BerHirt08]

- Our new result: $O(n \log n + k)$ [2)]

(actually: $O(n \log n + k/n^c)$ for any c - can probably be removed)

1) Amortized complexity: assumes large enough circuits

2) Requires not too large multiplicative depth

Tricks

Protocol makes use of known techniques:

- Shamir secret sharing [Sha79]
- Beaver's circuit randomization [Bea89]
- dispute control [BerHirt06]
- linear-time passively-secure multiplication [DamNie07]
- ...

and cumbersome fine-tuning, but crucially relies on **two new tricks**:

1. efficient batch verification for multiplication triples [3)]
   (to verify $c = a \cdot b$ **for many** shared triples (a, b, c) **in one go**)
2. efficient "mini MPC" for computing authentication tags

3) Independent work: similar trick used in [CraDamPas12], in setting of computational interactive proofs

Reconstruction in the Presence of Faults

secret: s

$f(X) = s + a_1 X + \ldots + a_t X^t$

shares: $s_1 = f(x_1)$ $\ldots$ $s_i = f(x_i)$ $\ldots$ $s_k = f(x_k)$ $\ldots$ $s_n = f(x_n)$

- **Problem:** how to reconstruct s if up to t shares are faulty?
- In case $n/3 \leq t < n/2$: impossible (without additional redundancy)
- **Idea** [RB89]: authenticate the shares

Reconstruction in the Presence of Faults

secret: s

$$f(X) = s + a_1 X + \ldots + a_t X^t$$

shares: $s_1 = f(x_1)$ $\ldots$ $s_i = f(x_i)$ $\ldots$ $s_k = f(x_k)$ $\ldots$ $s_n = f(x_n)$

$\tau_{11}\,,\ (\alpha_{11},\beta_{11})$
$\vdots$
$\tau_{1n}\,,\ (\alpha_{1n},\beta_{1n})$

$\ldots$

$\tau_{i1}\,,\ (\alpha_{i1},\beta_{i1})$
$\vdots$
$\tau_{ik}\,,\ (\alpha_{ik},\beta_{ik})$
$\vdots$
$\tau_{in}\,,\ (\alpha_{in},\beta_{in})$

$\tau_{k1}\,,\ (\alpha_{k1},\beta_{k1})$
$\vdots$
$\tau_{ki}\,,\ (\alpha_{ki},\beta_{ki})$
$\vdots$
$\tau_{kn}\,,\ (\alpha_{kn},\beta_{kn})$

$\ldots$

$\tau_{n1}\,,\ (\alpha_{n1},\beta_{n1})$
$\vdots$
$\tau_{nn}\,,\ (\alpha_{nn},\beta_{nn})$

$$\tau_{ik} = \alpha_{ki} \cdot s_i + \beta_{ki}$$

Problem #1: Blows up complexity!

Problem #2: Who **computes** the tag $\tau_{ik} = \alpha_{ki}\, s_i + \beta_{ki}$?

Solving Problem #1

- Authenticate large blocks of shares $s_i^1, \dots, s_i^L$ (for secrets $s^1, \dots, s^L$) via

$$\tau = \boldsymbol{\alpha} \cdot \boldsymbol{s}_i + \beta = \textstyle\sum_\ell \alpha^\ell \, s_i^\ell + \beta$$

  with key $\boldsymbol{\alpha} = (\alpha^1, \dots, \alpha^L)$ and β (actually: τ_{ki}, $\boldsymbol{\alpha}_{ki}$ and β_{ki}).

  For large L, efficiency loss due to β and τ becomes negligible.

- Use the same $\boldsymbol{\alpha} = (\alpha^1, \dots, \alpha^L)$ for different blocks $\boldsymbol{s}_i = (s_i^1, \dots, s_i^L)$.

  For many blocks, efficiency loss due to $\boldsymbol{\alpha}$ becomes negligible.

Solving Problem #2

Problem #2: Who computes tag $\tau = \alpha\, s_i + \beta$ (actually $\sum_\ell \alpha^\ell\, s_i^\ell + \beta$)?

Recall:

- P_k – who holds (α,β) – is not supposed to learn s_i
- P_i – who holds s_i – is not supposed to learn (α,β)
- dealer is not supposed to learn (α,β) – as he might be dishonest

Standard approach/solution:

- do a 2-level sharing: every s_i is re-shares into $s_{i1,\ldots,}s_{in}$
- sub-shares s_{ij} are authenticated

quadratic complexity ↯

- player P_i computes tags for sub-shares $s_{i1,\ldots,}s_{in}$ of s_i

Solving Problem #2

Problem #2: Who computes tag $\tau = \alpha\, s_i + \beta$ (actually $\sum_\ell \alpha^\ell\, s_i^\ell + \beta$)?

Recall:

- P_k - who holds (α,β) - is not supposed to learn s_i
- P_i - who holds s_i - is not supposed to learn (α,β)
- dealer is not supposed to learn (α,β) - as he might be dishonest

New approach: by means of a **MPC**

? ? ?

Appears hopeless:
just sharing the input, s_i, leads to quadratic complexity

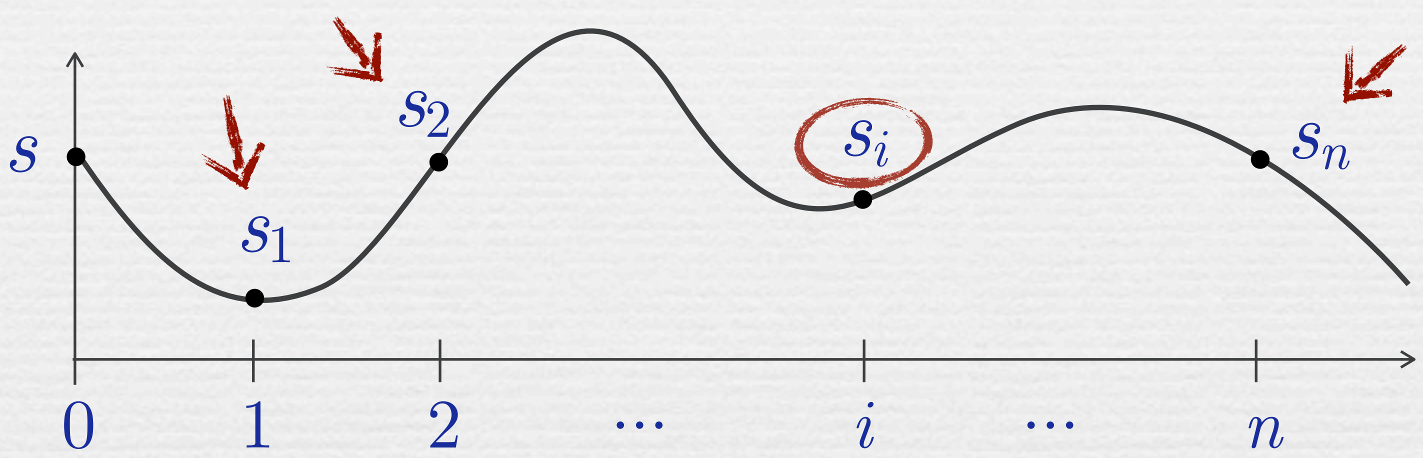
Good news:

- Circuit is very simple: multiplicative depth 1
- Don't need to worry about other inputs, α and β
- Dispute control framework => only need passive security
  (correctness can be verified by cut-and-choose)

Solving Problem #2

Solution: To not share the share s_i

Instead: use the remaining shares $(s_j)_{j\neq i}$ of s as shares of s_i



Fact:

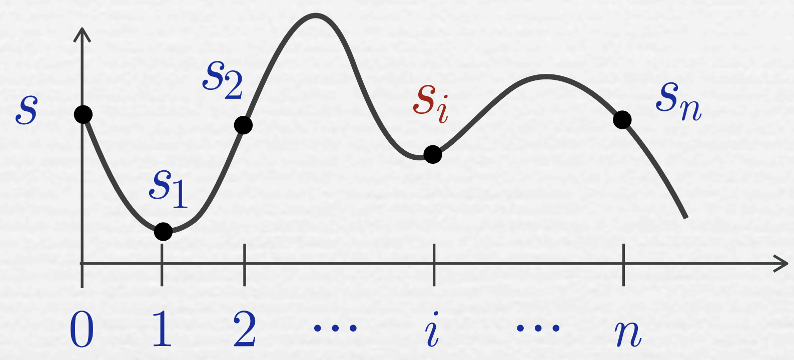
- any t of the shares $(s_j)_{j\neq i}$ give no info on s_i
- any $t{+}1$ of the shares $(s_j)_{j\neq i}$ reveal s_i

Thus: $(s_j)_{j\neq i}$ **is** a sharing of s_i, wrt. to a variant of Shamir's scheme
(where secret is evaluation of f at point i, rather than at 0)

Multiparty-Computing the Tag

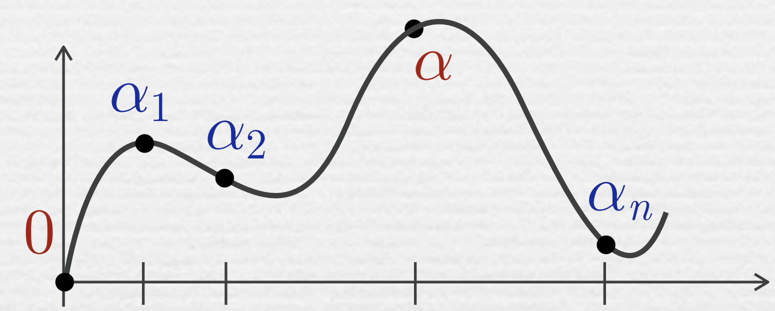
Protocol MINIMPC

- Given: shares $s_1,\ldots,s_i,\ldots,s_n$



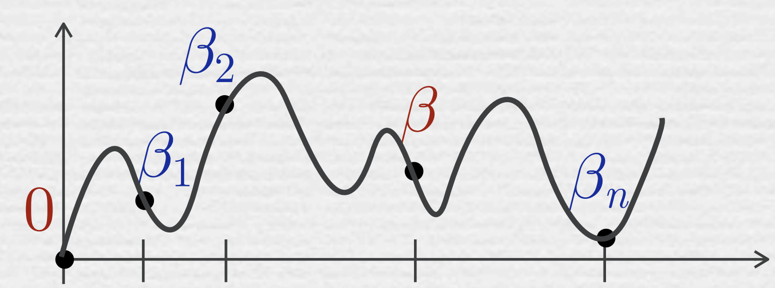
$\deg(f) = t$
$f(0) = s$

- P_k shares α as follows
  (P_i gets no share)



$\deg(g) = t$
$g(i) = \alpha$
$g(0) = 0$

- P_k shares β as follows
  (P_i gets no share)

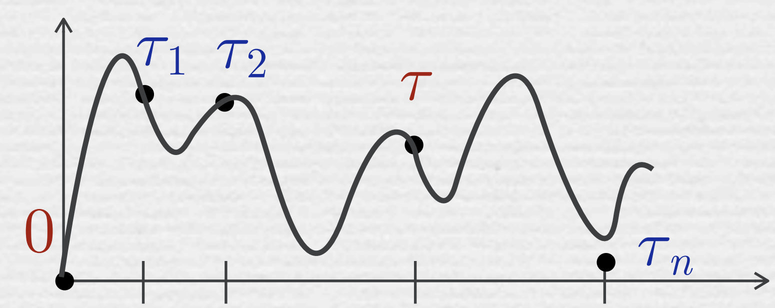


$\deg(h) = 2t$
$h(i) = \beta$
$h(0) = 0$

- every P_j ($j \neq i$) sends

$$\tau_j = \alpha_j s_j + \beta_j$$

  to P_i

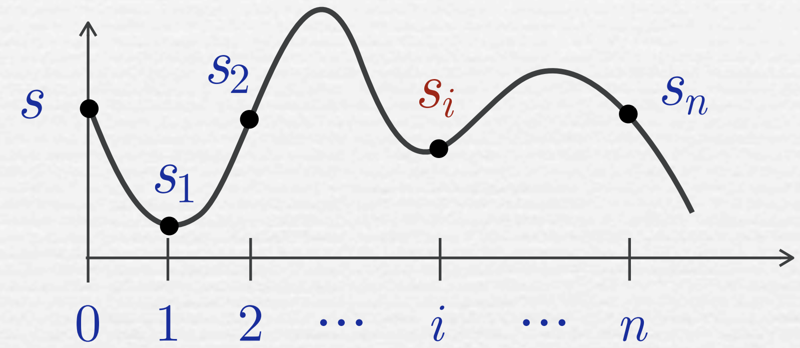


- P_i reconstructs $\tau = \alpha\, s_i + \beta$ from τ_j's

Multiparty-Computing the Tag

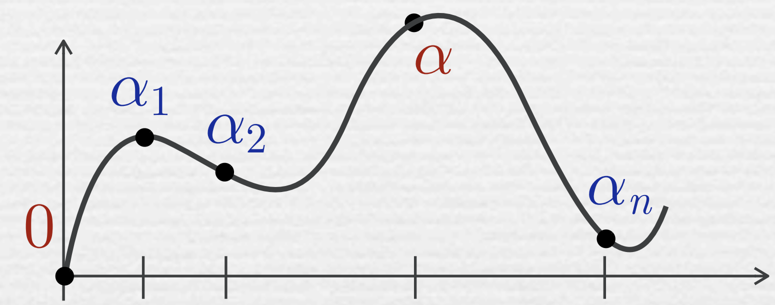
Protocol MINIMPC

- Given: shares $s_1,\ldots,s_i,\ldots,s_n$



$\deg(f) = t$
$f(0) = s$

- P_k shares α as follows
  (P_i gets no share)



$\deg(g) = t$
$g(i) = \alpha$
$g(0) = 0$

- P_k
  (P

$) = 2t$
$= \beta$
$= 0$

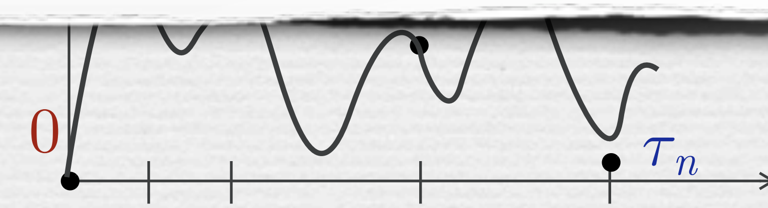
- ev

Note:

Adversary can learn α by corrupting t players $P_j \neq P_i$.
But α is of no use, if he does not corrupt P_i.

$\tau_j = \alpha_j s_j + \beta_j$

to P_i



- P_i reconstructs $\tau = \alpha\, s_i + \beta$ from τ_j's

Conclusion

- $\exists$ unconditionally-secure MPC with near-linear complexity
- There exist cases where MPC **improves** efficiency

- Open problems:
  - Improve circuit-independent part of the complexity: $O(n^7 k)$
  - Remove restriction on multiplicative depth of circuit
    (also present in the simpler $t < n/3$ setting)
  - What about non-threshold adversary structures?
    (Mini MPC crucially relies on Shamir's secret sharing scheme)