Efficient Proofs of Computational Integrity from Code-Based Interactive Oracle Proofs

Sarah Bordage

Project-team GRACE
LIX, Ecole Polytechnique, Institut Polytechnique de Paris
Inria Saclay Ile-de-France

GT Codes et Cryptographie
June 17, 2021
Verifiable Computing

Please, run program $F$ on input $x$ for me.

I want to quickly check if your result is correct.

**Powerful Prover**

On input $(F, x)$, output result $y$ and proof of correctness $\pi$.

**Weak Verifier**

On input $(F, x, y, \pi)$, accept iff $\pi$ is a valid proof for statement "$y = F(x)$".

- **Completeness**: Verifier $V$ always accepts honest proof.
- **Soundness**: For any malicious $\tilde{P}$, $\Pr[\tilde{P} \text{ convinces } V \text{ to accept incorrect statement}] = \text{negl}$.
- **Efficiency**: verification time, length of $\pi \ll$ time for computing $F(x)$.

*Remark*: sublinear time($V$) requires $|\text{description of the computation}| \ll |\text{running time of the computation}|$, $V$ should not “unroll” the computation.
A view of the “proofs-space” (by crypto assumptions)

<table>
<thead>
<tr>
<th>Year</th>
<th>CRHF, ROM</th>
<th>DLOG</th>
<th>KoE/AGM/GGM (pairing-based)</th>
<th>Group of unknown order</th>
</tr>
</thead>
<tbody>
<tr>
<td>2013</td>
<td></td>
<td></td>
<td>Pinocchio [PGHR]</td>
<td></td>
</tr>
<tr>
<td>2014</td>
<td></td>
<td></td>
<td>[BCTV]</td>
<td></td>
</tr>
<tr>
<td>2016</td>
<td>ZKBoo [GMO]</td>
<td>[BCCGP]</td>
<td>[Groth16] [GM]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SCI [BBC+]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2017</td>
<td>Ligero [AHIV]</td>
<td>Bulletproof [BBB+]</td>
<td>(ZK) vSQL [ZGK+]</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Hyrax [WTS+]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2018</td>
<td>Stark [BBHR]</td>
<td></td>
<td>vRAM [ZGK+]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Aurora [BCR+]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RedShift [KPV]</td>
<td></td>
<td>Marlin [CHM+]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Virgo [ZXZS]</td>
<td></td>
<td>Libra [XZZ+]</td>
<td></td>
</tr>
<tr>
<td>2020</td>
<td>Virgo++ [ZWZZ]</td>
<td></td>
<td>Mirage [KKPS]</td>
<td></td>
</tr>
</tbody>
</table>

Some implementations of succinct non-interactive arguments for general computations
1. IOP-based succinct non-interactive arguments

2. STARK Arithmetization

3. Reed-Solomon Proximity Testing
IOP-based succinct non-interactive arguments
Probabilistically Checkable Proofs

Given a relation $\mathcal{R}$ of instance-witness pairs $(x, w)$, denote $L_{\mathcal{R}} = \{x \mid \exists w, (x, w) \in \mathcal{R}\}$.

**Probabilistically checkable proof system (PCP)**

For verifiable computing, important measures are:

- $\text{time}(\mathcal{P})$,
- $\text{time}(\mathcal{V})$,
- query complexity $q$,
- proof length $l$.

**Completeness**: $\forall (x, w) \in \mathcal{R}, \exists \pi, \mathcal{V}^\pi(x) = 1$

**Soundness**: $\forall x \notin L_{\mathcal{R}}, \forall \tilde{\pi}, \Pr[\mathcal{V}^\pi(x) = 0] > 1/2$
PCP-based succinct interactive argument

\[ \mathcal{P}(x, w) \xleftarrow{} \text{computationally bounded} \]

CRHF \( H : \{0, 1\}^* \rightarrow \{0, 1\}^\kappa \)

\[ \pi \xleftarrow{} \mathcal{P}(x, w) \quad \text{root} = \text{MTree}_H(\pi) \]

\[ \text{paths} = (\text{MProofs}_H(\pi[k]))_{k \in Q} \]

Sample unif. at random query set \( Q \) of size \( q \)

Check Merkle proofs

\[ q = |Q|, l = |\pi| \]

From \( q \)-queries PCP of length \( l \):

- Communication = \( O_\kappa(q \log l) \),
- \( \text{time}(\mathcal{P}) = \text{time}(\mathcal{P}) + O_\kappa(l) \),
- \( \text{time}(\mathcal{V}) = \text{time}(\mathcal{V}) + O_\kappa(q \log l) \).

\[ \Rightarrow [\text{Kilian}'92] \text{PCP} (\mathcal{P}, \mathcal{V}) + \text{collision-resistant hashings} \]

\[ \leadsto \text{interactive argument} (\mathcal{P}, \mathcal{V}) \text{ for } \text{NP} \text{ with sublinear communication.} \]
PCP-based succinct non-interactive argument

\[ \text{\( \Pi(x, w) \) \hfill computationally bounded \hfill \( \forall(x) \)}} \]

\[ \text{RO \( H : \{0, 1\}^* \rightarrow \{0, 1\}^\kappa \)}} \]

\[ \pi \leftarrow \mathcal{P}(x, w) \]
\[ \text{root} = \text{MTree}_H(\pi) \]

\[ Q \leftarrow H(x || \text{root}) \]

\[ \text{paths} = (\text{MProofs}_H(\pi[k]))_{k \in Q} \]

\[ Q \leftarrow H(x || \text{root}) \]

Check Merkle proofs

Output 1 iff \( \forall^\pi(x) = 1 \)

\[ q = |Q|, l = |\pi| \]

From \( q \)-queries PCP of length \( l \):

- Communication = \( O_\kappa(q \log l) \),
- \( \text{time}(\Pi) = \text{time}(\mathcal{P}) + O_\kappa(l) \),
- \( \text{time}(\forall) = \text{time}(\forall) + O_\kappa(q \log l) \).

- [Kilian'92] PCP \( (\mathcal{P}, \forall) \) + collision-resistant hashings
  \( \leadsto \) interactive argument \( (\Pi, \forall) \) for \( \text{NP} \) with sublinear communication.

- [Micali'00] Kilian’s protocol + Fiat-Shamir paradigm \( (H \text{ random oracle}; Q \leftarrow H(x, \text{root})) \)
  \( \leadsto \) succinct non-interactive argument for \( \text{NP} \) in the Random Oracle Model.
Interactive Oracle Proofs

[Ben–Sasson-Chiesa-Spooner’16, Reingold-Rothblum-Rothblum’16]

**IOP system**

$(\mathcal{P}, \mathcal{V})$ is an IOP system for relation $\mathcal{R}$ with soundness error $\varepsilon$ if

**Completeness:**  
$\forall (x, w) \in \mathcal{R}, \Pr[\langle P(x, w), V^{\pi_1, \ldots, \pi_r}(x) \rangle = 1] = 1$.

**Soundness:**  
$\forall x \notin \mathcal{L}_{\mathcal{R}}, \forall \widetilde{P}, \Pr[\langle P(x, w), V^{\pi_1, \ldots, \pi_r}(x) \rangle = 1] \leq \varepsilon$.

Proof length $l = \sum |\pi_i|$  
Query complexity $q = \text{number of queries to } \pi_1, \ldots, \pi_r$
IOP-based succinct non-interactive argument [Ben–Sasson-Chiesa-Spooner’16]

CRHF $H : \{0, 1\}^n \rightarrow \{0, 1\}^\kappa$

For each round $i$

$\pi_i \leftarrow \mathcal{P}(x, w)$

$\text{root}_i = \text{MTree}_H(\pi_i)$

Sample unif. at random query set $Q_i$

$\pi_i \mid Q_i \rightarrow \text{paths}_i = (\text{MProofs}_H(\pi_i[k]))_{k \in Q_i}$

Check Merkle proofs

Output 1 iff $\forall \pi_1, \ldots, \pi_r(x) = 1$

$q = \sum |Q_i|, l = \sum |\pi_i|$

From $q$-queries IOP with length $l$:

- Communication = $O_\kappa(q \log l)$,
- time($\mathcal{P}$) = time($\mathcal{P}$) + $O_\kappa(l)$,
- time($\mathcal{V}$) = time($\mathcal{V}$) + $O_\kappa(q \log l)$.

Interactive argument from Merkle Trees

IOP system ($\mathcal{P}, \mathcal{V}$) + hash functions $\implies$ succinct interactive argument ($\mathcal{P}, \mathcal{V}$)

information theoretic

collision-resistant
IOP-based succinct non-interactive argument [Ben–Sasson-Chiesa-Spooner’16]

**CRHF** $H : \{0, 1\}^n \rightarrow \{0, 1\}^\kappa$

For each round $i$

- $\pi_i \leftarrow \mathcal{P}(x, w)$
- $\text{root}_i = \text{MTree}_H(\pi_i)$

Sample unif. at random query set $Q_i$

- $Q_i \leftarrow \mathcal{P}(x, w)
- \text{paths}_i = (\text{MProofs}_H(\pi_i[k]))_{k \in Q_i}$

Check Merkle proofs

Output 1 iff $\mathcal{V}^{\pi_1, \ldots, \pi_r}(x) = 1$

From $q$-queries IOP with length $l$:
- Communication $= O_\kappa(q \log l)$,
- $\text{time}(\mathcal{P}) = \text{time}(\mathcal{P}) + O_\kappa(l)$,
- $\text{time}(\mathcal{V}) = \text{time}(\mathcal{V}) + O_\kappa(q \log l)$.

Remove interaction: $Q_i$ deduced from $RO(x||\text{root}_1||\ldots||\text{root}_{i-1}||Q_0||\ldots||Q_{i-1})$

**Non-interactive argument from Merkle Trees + Fiat-Shamir**

**IOP system** $\mathcal{P}, \mathcal{V}$ + hash functions $\rightarrow$ succinct non-interactive argument $(\mathcal{P}, \mathcal{V})$
**Computational integrity language** $\mathcal{L}$:
Instances $(F, x, y, T)$ such that running program $F$ within $T$ cycles on public input $x$ and auxiliary (secret) witness $w$ leads to output $y$.

**STARK (Scalable Transparent ARGument of KNOWledge)** [Ben-Sasson-Bentov-Horesh-Riabzev'18]

$\rightarrow$ non-interactive argument of knowledge for “machine computations”, with or without zero-knowledge

<table>
<thead>
<tr>
<th>Prover</th>
<th>Verifier</th>
<th>Communication complexity</th>
<th>Setup</th>
<th>Post-Quantum</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O_\kappa(T \log^2 T) \cdot H$</td>
<td>$O_\kappa(\log^2 T) \cdot H$</td>
<td>$O_\kappa(\log^2 T)$</td>
<td>transparent</td>
<td>yes</td>
</tr>
</tbody>
</table>

Construct an IOP system $(\mathcal{P}, \mathcal{V})$ for $\mathcal{L}$ with:

$$
time(\mathcal{P}) = O(T \log^2 T) \quad \text{time}(\mathcal{V}) = \~O(n) + O(\log(T)), \quad l = O(T \log T), \quad q = O(\log T),$$

then apply [BCS'16] transformation.

**Applications:**
Allows verification of multiple programs in a single proof (StarkEx, Cairo).
One can build PQ signatures from ZK-STARKs (see Ziggy STARK).
STARK Arithmetization:
From computational integrity to low-degree testing
Toy example

**Toy program**

- **Public input:** $x \in \mathbb{F}$, steps $T$ ($T \ll |\mathbb{F}|$)
- **Private output:** $K \in \mathbb{F}^T$, $K = (K_0, K_1, \ldots, K_{T-1})$

**Program** $F(x, T, K)$:

For $i = 0$ to $T - 1$:

\[ x \leftarrow x^3 + K_i \]

return $x$

**Execution trace**

<table>
<thead>
<tr>
<th>$x_0$</th>
<th>$K_0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>$K_1$</td>
</tr>
<tr>
<td>$x_2$</td>
<td>$K_2$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$x_{T-2}$</td>
<td>$K_{T-2}$</td>
</tr>
<tr>
<td>$x_{T-1}$</td>
<td>$K_{T-1}$</td>
</tr>
<tr>
<td>$x_T$</td>
<td>0</td>
</tr>
</tbody>
</table>

Valid iff

\[
\begin{align*}
    x_0 &= x \\
    x_T &= y \\
    x_{i+1} &= x_i^3 + K_i
\end{align*}
\]

**Computational integrity statement for ZK-STARK:**

“Given program $F$, public input $(x, T)$ and public output $y$, I know secret $K$ such that $F(x, T, K) = y$.”
STARK: Toy example

**Toy program**

- **Public input**: \( x \in \mathbb{F}, \) steps \( T (T \ll |\mathbb{F}|) \)
- **Private output**: \( K \in \mathbb{F}^T, K = (K_0, K_1, \ldots, K_{T-1}) \)

**Program** \( F(x, T, K) \):

For \( i = 0 \) to \( T - 1 \):

\[
\begin{align*}
  x &\leftarrow x^3 + K_i \\
\end{align*}
\]

return \( x \)

---

**Execution trace**

| \( x_0 \) | \( K_0 \) |
| \( x_1 \) | \( K_1 \) |
| \( x_2 \) | \( K_2 \) |
| ... | ... |
| \( x_{T-2} \) | \( K_{T-2} \) |
| \( x_{T-1} \) | \( K_{T-1} \) |
| \( x_T \) | 0 |

Valid iff

\[
\begin{align*}
  x_0 &= x \\
  x_T &= y \\
  x_{i+1} &= x_i^3 + K_i
\end{align*}
\]

**Computational integrity statement for ZK-STARK:**

“Given program \( F \), public input \( (x, T) \) and public output \( y \), I know secret \( K \) such that \( F(x, T, K) = y \).”

... Today, we’ll discuss only the computational integrity part.

\( \rightarrow \) “Given program \( F \), public input \( (x, T) \) and public output \( y \), there exists \( K \) such that \( F(x, T, K) = y \).”
\( \mathcal{P} \) and \( \mathcal{V} \) agree on \( \omega \in \mathbb{F}^\times \) of order \( T + 1 \), which defines \( G := \langle \omega \rangle \). They also define:

- “Contraint polynomial” \( C(X_0, X_1, Y_0) = Y_0 - (X_0^3 + X_1) \)
- “Boundaries” polynomial \( B \in \mathbb{F}[X]_{\leq 1} \) such that \( B(1) = x \) and \( B(\omega^T) = y \)
- Vanishing polynomial \( Z(X) = \prod_{i=0}^{T-1} (X - \omega^i) = \frac{X^{T+1} - 1}{X - \omega^T} \)

\( \mathcal{P} \) interpolates \( P_0, P_1 \in \mathbb{F}[X]_{\leq T} \) such that for all \( i \in \{0, ..., T\} \), \( P_0(\omega^i) = x_i \) and \( P_1(\omega^i) = K_i \).

Valid execution trace \iff\ \begin{align*}
  x_0 &= x, \\
  x_T &= y, \\
  C(x_i, K_i, x_{i+1}) &= 0, & \text{for } 0 \leq i < T
\end{align*}
Toy example: Arithmetization I

\( \mathcal{P} \) and \( \mathcal{V} \) agree on \( \omega \in \mathbb{F}^\times \) of order \( T + 1 \), which defines \( G := \langle \omega \rangle \). They also define:

- “Contraint polynomial” \( C(X_0, X_1, Y_0) = Y_0 - (X_0^3 + X_1) \)
- “Boundaries” polynomial \( B \in \mathbb{F}[X]_{\leq 1} \) such that \( B(1) = x \) and \( B(\omega^T) = y \)
- Vanishing polynomial \( Z(X) = \prod_{i=0}^{T-1} (X - \omega^i) = \frac{X^{T+1} - 1}{X - \omega^T} \)

\( \mathcal{P} \) interpolates \( P_0, P_1 \in \mathbb{F}[X]_{\leq T} \) such that for all \( i \in \{0, ..., T\} \), \( P_0(\omega^i) = x_i \) and \( P_1(\omega^i) = K_i \).

Valid execution trace \( \iff \begin{cases} x_0 = x, \\ x_T = y, \\ C(x_i, K_i, x_{i+1}) = 0, \text{ for } 0 \leq i < T \end{cases} \)

\( \iff \begin{cases} P_0(\omega^0) = B(\omega^0), \\ P_0(\omega^T) = B(\omega^T), \\ C(P_0(\omega^i), P_1(\omega^i), P_0(\omega^{i+1})) = 0, \text{ for } 0 \leq i < T \end{cases} \)
Toy example: Arithmetization I

\( \mathcal{P} \) and \( \mathcal{V} \) agree on \( \omega \in \mathbb{F}^\times \) of order \( T + 1 \), which defines \( G := \langle \omega \rangle \). They also define:

- “Contraint polynomial” \( C(X_0, X_1, Y_0) = Y_0 - (X_0^3 + X_1) \)
- “Boundaries” polynomial \( B \in \mathbb{F}[X]_{\leq 1} \) such that \( B(1) = x \) and \( B(\omega^T) = y \)
- Vanishing polynomial \( Z(X) = \prod_{i=0}^{T-1} (X - \omega^i) = \frac{X^{T+1} - 1}{X - \omega^T} \)

\( \mathcal{P} \) interpolates \( P_0, P_1 \in \mathbb{F}[X]_{\leq T} \) such that for all \( i \in \{0, ..., T\} \), \( P_0(\omega^i) = x_i \) and \( P_1(\omega^i) = K_i \).

**Valid execution trace**

\[ \iff \begin{cases} x_0 = x, \\ x_T = y, \\ C(x_i, K_i, x_{i+1}) = 0, \text{ for } 0 \leq i < T \end{cases} \]

\[ \iff \begin{cases} P_0(\omega^0) = B(\omega^0), \\ P_0(\omega^T) = B(\omega^T), \\ C(P_0(\omega^i), P_1(\omega^i), P_0(\omega^{i+1})) = 0, \text{ for } 0 \leq i < T \end{cases} \]

\[ \iff \begin{cases} (X - 1)(X - \omega^T) \text{ divides } P_0(X) - B(X) \\ Z(X) \text{ divides } C(P_0(X), P_1(X), P_0(hX)) \end{cases} \]
Valid execution trace $\iff$

\[
\begin{align*}
Q_0(X) &= \frac{P_0(X) - B(X)}{(X-1)(X-\omega^T)} \in \mathbb{F}_q[X]_{\leq T-2} \\
Q_1(X) &= \frac{C(P_0(X), P_1(X), P_0(hX))}{Z(X)} \in \mathbb{F}_q[X]_{\leq 2T}
\end{align*}
\]
Valid execution trace $\iff\begin{cases} Q_0(X) = \frac{P_0(X) - B(X)}{(X-1)(X-\omega^T)} \in \mathbb{F}_q[X] \leq T - 2 \\ Q_1(X) = \frac{C(P_0(X), P_1(X), P_0(hX))}{Z(X)} \in \mathbb{F}_q[X] \leq 2T \end{cases}$

**Reed-Solomon encoding**

$\text{RS}[\mathbb{F}, D, k] \subset \mathbb{F}^D$: evaluations on $D \subset \mathbb{F}$ of polynomials of degree $< k$.

We assume $|D| \geq 2(T + 1)$, $|D| = \Theta(T)$, and $D \cap G = \emptyset$.

$\mathcal{P}$ sends oracle functions $f_0, f_1, g_0, g_1 \in \mathbb{F}^D$, supposedly evaluations on $D$ of $P_0(X), P_1(X), Q_0(X), Q_1(X)$, respectively.

Valid execution trace $\iff\begin{cases} f_0, f_1 \in \text{RS}[\mathbb{F}, D, T + 1] \\ g_0 \in \text{RS}[\mathbb{F}, D, T - 1] \\ g_1 \in \text{RS}[\mathbb{F}, D, 2T + 1] \end{cases}$
Toy example: Arithmetization III

Assume $f_0, f_1 \in \text{RS}[\mathbb{F}, D, T + 1]$. Then,

- $(x_0 \neq x \text{ or } x_T \neq y) \implies \Delta(g_0, \text{RS}[\mathbb{F}, D, T - 1]) \geq 1 - \frac{T + 1}{|D|}$
- $\exists i, x_{i+1} \neq x_i^3 + K_i \implies \Delta(g_1, \text{RS}[\mathbb{F}, D, 2T + 1]) \geq 1 - \frac{3T + 1}{|D|}$

Arithmetization:

- **valid execution trace** $\leadsto$ **RS codewords**
- **incorrect execution trace** $\leadsto$ **words far from any RS codewords**
Assume $f_0, f_1 \in \text{RS}[\mathbb{F}, D, T + 1]$. Then,

- $(x_0 \neq x$ or $x_T \neq y) \implies \Delta(g_0, \text{RS}[\mathbb{F}, D, T - 1]) \geq 1 - \frac{T + 1}{|D|}$ \quad $\Delta$ relative Hamming distance
- $\exists i, x_{i+1} \neq x_i^3 + K_i \implies \Delta(g_1, \text{RS}[\mathbb{F}, D, 2T + 1]) \geq 1 - \frac{3T + 1}{|D|}$

Arithmetization: \L\left\{\begin{array}{l}
\text{valid execution trace $\sim$ RS codewords} \\
\text{incorrect execution trace $\sim$ words far from any RS codewords}
\end{array}\right.$

**Consistency at a random location**

$\mathcal{V}$ samples $s \in D$ and requests $f_0(s), f_1(s), f_0(\omega \cdot s), g_0(s), g_1(s)$, computes $B(s)$ and $Z(s)$ (in $O(\log T)$ $\mathbb{F}$-ops), then checks:

1. $g_0(s) \cdot (s - 1) \cdot (s - \omega^T) \stackrel{?}{=} f_0(s) - B(s)$ \quad soundness error $\leq \frac{T}{|D|}$
2. $g_1(s) \cdot Z(s) \stackrel{?}{=} C(f_0(s), f_1(s), f_0(\omega \cdot s))$ \quad soundness error $\leq \frac{3T}{|D|}$

$\mathcal{V}$ rejects if one of the test fails.

**Low-degree compliance**

Instead of RS membership test: $\mathcal{V}$ must be able to test proximity to RS code with $O(\log T)$ queries.
Reed-Solomon Proximity Testing
### Reed-Solomon Proximity Testing

#### Reed-Solomon proximity testing

<table>
<thead>
<tr>
<th>Input:</th>
<th>a code $\text{RS}[^F, D, k]$, a parameter $\delta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input oracle:</td>
<td>$f : D \to ^F$</td>
</tr>
<tr>
<td>Completeness:</td>
<td>If $f \in \text{RS}[^F, D, k]$, then the test always accepts.</td>
</tr>
<tr>
<td>Soundness:</td>
<td>If $\Delta \left(f, \text{RS}[^F, D, k]\right) &gt; \delta$, then the test accepts with probability $\leq \text{err}(\delta)$.</td>
</tr>
</tbody>
</table>

$\Delta$ relative Hamming distance
Reed-Solomon Proximity Testing

Reed-Solomon proximity testing

Input: a code $RS[F, D, k]$, a parameter $\delta$
Input oracle: $f : D \rightarrow F$
Completeness: If $f \in RS[F, D, k]$, then the test always accepts.
Soundness: If $\Delta(f, RS[F, D, k]) > \delta$, then the test accepts with probability $\leq \text{err}(\delta)$.
$\Delta$ relative Hamming distance

Naive proximity test

1. Query $k + 1$ entries of $f \in F^D : f(x_0), \ldots, f(x_k)$,
2. Compute poly $P(X)$ of degree $< k$ by interpolating $\{(x_i, f(x_i)); 0 \leq i \leq k - 1\}$
3. Tester accepts iff $P(x_k) = f(x_k)$.
   Soundness: $\forall$ accepts with proba $\Pr_{x_k}[P(x_k) \neq f(x_k)] \leq 1 - \Delta(f, RS[F, D, k])$.
   Problem: query complexity $k + 1$ is linear in $|D|$ with $|D| = \Theta(k)$.

How to achieve logarithmic verification time? $\forall$ will need some auxiliary proof from $\mathcal{P}$...
We also want a fast prover (linear in $|D|$) $\rightsquigarrow$ IOP of Proximity (IOPP) for RS code.
### Reed-Solomon IOP of Proximity

**RS IOP of Proximity**

- **Input:** a code $\text{RS}[F, D, k]$, a parameter $\delta$
- **Input oracle:** $f : D \rightarrow F$
- **Completeness:** If $f \in \text{RS}[F, D, k]$, then $\exists \mathcal{P} \Pr[\langle \mathcal{P}, \mathcal{V} \rangle = 1] = 1$
- **Soundness:** If $\Delta \left( f, \text{RS}[F, D, k] \right) > \delta$, then $\forall \tilde{\mathcal{P}} \Pr[\langle \tilde{\mathcal{P}}, \mathcal{V} \rangle = 1] \leq \text{err}(\delta)$

$\Delta$ relative Hamming distance

---

### Fast Reed-Solomon IOPP: FRI Protocol

[Ben–Sasson-Bentov-Horesh-Riabzev'18]

- # rounds $< \log |D|$
- # queries $< 2\log |D|$
- prover time $< 6|D|$
- verifier time $< 21\log |D|$
- proof length $< |D|/3$
- soundness “$\approx 1 - \delta$”
Halving the size of the problem by folding

Assume there exists \( \omega \in \mathbb{F}^\times \) of order a large power of 2.
Given \( k = 2^r \) and evaluation domain \( D = \langle \omega \rangle \) of size \( n = |D| \) and closed under negation

**How to check proximity of \( f_0 : D \to \mathbb{F} \) to the code \( \text{RS}[\mathbb{F}, D, k] \)?**

Define sequence of RS codes: \( \text{RS}_i := \text{RS}[\mathbb{F}, D_i, k/2^i] \) with \( D_0 = D \) and \( D_i := \langle \omega^{2^i} \rangle \)
(same rate \( \rho = k/n \))
Halving the size of the problem by folding,

Assume there exists $\omega \in \mathbb{F}^\times$ of order a large power of 2.
Given $k = 2^r$ and evaluation domain $D = \langle \omega \rangle$ of size $n = |D|$ and closed under negation.

**How to check proximity of $f_0 : D \rightarrow \mathbb{F}$ to the code $\text{RS}[\mathbb{F}, D, k]$?**

Define sequence of RS codes: $\text{RS}_i := \text{RS}[\mathbb{F}, D_i, k/2^i]$ with $D_0 = D$ and $D_i := \langle \omega^{2^i} \rangle$ (same rate $\rho = k/n$)

**For each round $i$, reduce proximity to $\text{RS}_i$ to $\text{RS}_{i+1}$:**

- Split $f_i(X)$ into $u_0, u_1$, such that $f_i(X) = u_0(X^2) + X u_1(X^2)$
- For $\alpha \in \mathbb{F}$, define $\text{FOLD}_i[\alpha] : D_{i+1} \rightarrow \mathbb{F}$ by $\text{FOLD}_i[\alpha] (y) = u_0(y) + \alpha \cdot u_1(y)$

**Completeness**

For all $\alpha \in \mathbb{F}$, $\text{FOLD}_i[\alpha] \subseteq \text{RS}_{i+1}$
Halving the size of the problem by folding

Assume there exists $\omega \in \mathbb{F}^\times$ of order a large power of $2$.
Given $k = 2^r$ and evaluation domain $D = \langle \omega \rangle$ of size $n = |D|$ and closed under negation

**How to check proximity of $f_0 : D \to \mathbb{F}$ to the code $\text{RS}[\mathbb{F}, D, k]$?**

Define sequence of RS codes: $\text{RS}_i := \text{RS}[\mathbb{F}, D_i, k/2^i]$ with $D_0 = D$ and $D_i := \langle \omega^{2^i} \rangle$
(same rate $\rho = k/n$)

For each round $i$, reduce proximity to $\text{RS}_i$ to $\text{RS}_{i+1}$:

- Split $f_i(X)$ into $u_0, u_1$, such that $f_i(X) = u_0(X^2) + X u_1(X^2)$
- For $\alpha \in \mathbb{F}$, define $\text{FOLD} [f_i, \alpha] : D_{i+1} \to \mathbb{F}$ by $\text{FOLD} [f_i, \alpha] (y) = u_0(y) + \alpha \cdot u_1(y)$

**Completeness**

For all $\alpha \in \mathbb{F}$, $\text{FOLD} [\text{RS}_i, \alpha] \subseteq \text{RS}_{i+1}$

Observe: for all $x \in D_i$,

$$\text{FOLD} [f_i, \alpha] (x^2) = \frac{f_i(x) + f_i(-x)}{2} + \frac{\alpha f_i(x) - f_i(-x)}{2x}.$$  

**Local computability**

For any $y \in D_{i+1}$, compute $\text{FOLD} [f_i, \alpha] (y)$ with only 2 queries to $f_i$.  

\(17/24\)
Folding preserves distance to the code

Let $\varepsilon = \varepsilon(|F|, n, \rho) = O_{\rho} \left( \frac{n^2}{|F|} \right)$.

**Distance preservation**

Let $\delta < 1 - \sqrt{\rho}$. If $\Delta(f_i, RS_i) > \delta$, then

$$\Pr_{\alpha \in F} [\Delta(Fold [f_i, \alpha], RS_{i+1}) < \delta] < \varepsilon$$

Proof relies on:

**Theorem** [Ben-Sasson-Carmon-Ishai-Kopparty-Saraf'20]

Let $C \subseteq F^D$ be a RS code of rate $\rho$ and let $u_0, u_1 : D \to F$. For any $\delta < 1 - \sqrt{\rho}$, if

$$\Pr_{\alpha \in F} [\Delta(u_0 + \alpha u_1, C) < \delta] > 1 - \varepsilon,$$

then, there exist $c_0, c_1 \in C$ and $T \subset D$ such that

- $|T| \geq (1 - \delta) |D|$
- $u_0|_T = c_0|_T$ and $u_1|_T = c_1|_T$
Folding preserves distance to the code - proof sketch

**Distance preservation**

Let $\delta < 1 - \sqrt{\rho}$. If $\Delta(f_i, RS_i) > \delta$, then

$$\Pr_{\alpha \in \mathbb{F}} [\Delta(Fold[f_i, \alpha], RS_{i+1}) < \delta] < \varepsilon$$

**Proof sketch:**

Idea: assume $Fold[f_i, \alpha]$ is $\delta$-close of $RS_{i+1}$ for many $\alpha$’s,

$\leadsto$ reconstruct a codeword $v \in RS_i$ which is $\delta$-close to $f_i$. 

Distance preservation

Let $\delta < 1 - \sqrt{\rho}$. If $\Delta(f_i, \text{RS}_i) > \delta$, then

$$\Pr_{\alpha \in \mathbb{F}}[\Delta(\text{FOLD}[f_i, \alpha], \text{RS}_{i+1}) < \delta] < \epsilon$$

Proof sketch:

Idea: assume $\text{FOLD}[f_i, \alpha]$ is $\delta$-close of $\text{RS}_{i+1}$ for many $\alpha$'s, 

$\leadsto$ reconstruct a codeword $v \in \text{RS}_i$ which is $\delta$-close to $f_i$.

- Apply BCIKS'20 on $\text{FOLD}[f_i, \alpha] = u_0 + \alpha u_1 \implies \exists v_0, v_1 \in \text{RS}_{i+1}$ such that

$$T = \{y \in D_{i+1} \mid u_0(y) = v_0(y) \text{ AND } u_1(y) = v_1(y)\}$$

is of size $|T| \geq (1 - \delta) |D_{i+1}|$,

- Consider polynomial $r(X) = v_0(X^2) + Xv_1(X^2)$ of degree $< k/2^i$,

- Then, the evaluation of $r(X)$ on $D_i$ agrees with $f$ on the set $\{x \in D_i \mid x^2 \in T\}$ of size $\geq (1 - \delta) |D_i|$. 
FRI Protocol: COMMIT phase

Honest prover computes:

\[ f_1 = \text{FOLD}[f_0, \alpha_0] \]
\[ f_2 = \text{FOLD}[f_1, \alpha_1] \]
\[ \vdots \]
\[ f_r = \text{FOLD}[f_{r-1}, \alpha_{r-1}] \]
FRI Protocol: QUERY phase

**Global consistency test:**

Sample \( s \in D_0 \) and check

\[ f_1(s^2) \overset{?}{=} \text{FOLD } [f_0, \alpha_0] (s^2) \]

\[ f_2(s^4) \overset{?}{=} \text{FOLD } [f_1, \alpha_1] (s^4) \]

\[ \vdots \]

\[ f_r(s^{2^r}) \overset{?}{=} \text{FOLD } [f_{r-1}, \alpha_{r-1}] (s^{2^r}) \]

**Final test:** \( f_r \notin RS_r \)

output: 1 or 0
FRI Protocol: Soundness

What can go wrong?

- **COMMIT:**
  Event $\text{BAD}_i = \text{“at round } i, \text{ FOLD } [\cdot, \alpha_i] \text{ does not preserve distance to the code.”}\]

- **QUERY:** chosen queried path does not catch any errors.

\[
\Pr [\mathcal{V} \text{ accepts}] \leq \Pr_{\alpha_i \in F} \left[ \bigcup_{i=0}^{r-1} \text{BAD}_i \right] + \Pr_{s \in D} \left[ \mathcal{V} \text{ accepts } | \bigcap_{i=0}^{r-1} \overline{\text{BAD}_i} \right]
\]

\[
\leq \text{err}_{\text{commit}} + \text{err}_{\text{query}}
\]
FRI Protocol: Soundness

What can go wrong?

- **COMMIT:**
  Event $\text{BAD}_i$ = “at round $i$, FOLD $\cdot, \alpha_i$ does not preserve distance to the code.”

- **QUERY:** chosen queried path does not catch any errors.

\[
\Pr[\mathcal{V} \text{ accepts}] \leq \Pr_{\alpha_i \in \mathbb{F}} \left[ \bigcup_{i=0}^{r-1} \text{BAD}_i \right] + \Pr_{s \in D} \left[ \mathcal{V} \text{ accepts } | \bigcap_{i=0}^{r-1} \text{BAD}_i \right]
\]

**Theorem: Soundness** [Ben–Sasson-Kopparty-Saraf’18, Ben–Sasson-Carmon-Ishai-Kopparty-Saraf’20]

For any $\delta < 1 - \sqrt{\rho}$, if $\Delta(f, RS[\mathbb{F}, D, k]) > \delta$, then $\mathcal{V}$ accepts with proba at most

\[
\text{err}(\delta) < \text{err}_{\text{commit}} + (\text{err}_{\text{query}})^l
\]

\[
< r \cdot \varepsilon(|\mathbb{F}|, n, \rho) + (1 - \delta)^l
\]

after $l$ repetitions of the QUERY phase.
Let $\delta_i = \min \left( \Delta(f_i, RS_i), 1 - \sqrt{\rho} \right)$.

**Bounding $err_{commit}$:**

$BAD_i = \Delta(F/\!o.\!sc/\!l.\!sc/\!d.\!sc \left[ f_i, \alpha_i \right], RS_{i+1}) < \delta_i$.

By distance preservation: $\Pr[BAD_i] \leq \varepsilon$, thus $\Pr \left[ \bigcup_{i=0}^{r-1} BAD_i \right] \leq r \cdot \varepsilon$.
Let $\delta_i = \min \left( \Delta(f_i, RS_i), 1 - \sqrt{\rho} \right)$.

**Bounding $\text{err}_{\text{commit}}$:**

$\text{BAD}_i = "\Delta(\text{FOLD} [f_i, \alpha_i], RS_{i+1}) < \delta_i".$

By distance preservation: $\Pr[\text{BAD}_i] \leq \varepsilon$, thus $\Pr \left[ \bigcup_{i=0}^{r-1} \text{BAD}_i \right] \leq r \cdot \varepsilon$.

**Bounding $\text{err}_{\text{query}}$:**

**Suppose no $\text{BAD}_i$ occurs.**

Consider graph with vertices $D_0 \cup D_1 \cup \cdots \cup D_r$, and we’ll say that:

- $y \in D_{i+1}$ is **green** if $f_{i+1}(y) = \text{FOLD} [f_i, \alpha_i](y)$
- $y \in D_{i+1}$ is **red** if $f_{i+1}(y) \neq \text{FOLD} [f_i, \alpha_i](y)$

$\forall$ accepts iff every vertices of the queried path are green.
Let \( \delta_i = \min \left( \Delta(f_i, RS_i), 1 - \sqrt{\rho} \right) \).

**Bounding \( \text{err}_{\text{commit}} \):**

\( \text{BAD}_i = \text{"} \Delta(\text{FOLD}[f_i, \alpha_i], RS_{i+1}) < \delta_i \text{"}. \)

By distance preservation: \( \Pr[\text{BAD}_i] \leq \varepsilon \),
thus \( \Pr \left[ \bigcup_{i=0}^{r-1} \text{BAD}_i \right] \leq r \cdot \varepsilon \).

**Bounding \( \text{err}_{\text{query}} \):**

**Suppose no \( \text{BAD}_i \) occurs.**

Consider graph with vertices \( D_0 \cup D_1 \cup \cdots \cup D_r \), and we'll say that:

- \( y \in D_{i+1} \) is **green** if \( f_{i+1}(y) = \text{FOLD}[f_i, \alpha_i](y) \)
- \( y \in D_{i+1} \) is **red** if \( f_{i+1}(y) \neq \text{FOLD}[f_i, \alpha_i](y) \)

\( \leadsto V \) accepts iff every vertices of the queried path are green.

\[
\Pr[V \text{ accepts}] = \frac{6}{16} = \frac{3}{8}
\]
Let $\delta_i = \min \left( \Delta(f_i, RS_i), 1 - \sqrt{\rho} \right)$.

**Bounding $\text{err}_{\text{commit}}$:**
BAD$_i = \"\Delta(\text{FOLD} [f_i, \alpha_i], RS_{i+1}) < \delta_i\"$.

By distance preservation: $\Pr[\text{BAD}_i] \leq \varepsilon$, thus $\Pr\left[ \bigcup_{i=0}^{r-1} \text{BAD}_i \right] \leq r \cdot \varepsilon$.

**Bounding $\text{err}_{\text{query}}$:**

**Suppose no BAD$_i$ occurs.**
Consider graph with vertices $D_0 \cup D_1 \cup \cdots \cup D_r$, and we’ll say that:

- $y \in D_{i+1}$ is **green** if $f_{i+1}(y) = \text{FOLD} [f_i, \alpha_i](y)$
- $y \in D_{i+1}$ is **red** if $f_{i+1}(y) \neq \text{FOLD} [f_i, \alpha_i](y)$

$\mathcal{V}$ accepts iff every vertices of the queried path are green.

$\Pr[\mathcal{V} \text{ accepts}] = \frac{6}{16} = \frac{3}{8}$
Soundness analysis I

Let \( \delta_i = \min \left( \Delta(f_i, RS_i), 1 - \sqrt{\rho} \right) \).

**Bounding \( \text{err}_{\text{commit}} \):**

\( \text{BAD}_i = \text{“} \Delta(\text{FOLD}[f_i, \alpha_i], RS_{i+1}) < \delta_i \text{”} \).

By distance preservation: \( \Pr[\text{BAD}_i] \leq \varepsilon \),

thus \( \Pr\left[ \bigcup_{i=0}^{r-1} \text{BAD}_i \right] \leq r \cdot \varepsilon \).

**Bounding \( \text{err}_{\text{query}} \):**

**Suppose no \( \text{BAD}_i \) occurs.**

Consider graph with vertices \( D_0 \cup D_1 \cup \cdots \cup D_r \), and we’ll say that:

- \( y \in D_{i+1} \) is **green** if \( f_{i+1}(y) = \text{FOLD}[f_i, \alpha_i](y) \)
- \( y \in D_{i+1} \) is **red** if \( f_{i+1}(y) \neq \text{FOLD}[f_i, \alpha_i](y) \)

\( \Downarrow \) \( \mathcal{V} \) accepts iff every vertices of the queried path are green.

\[ \Pr[\mathcal{V} \text{ accepts}] = \frac{6}{16} = \frac{3}{8} \]

Modify the entries of \( f_1, \ldots, f_{r-1} \) to keep only the **last red vertex** along the path.

Modification process does not affect **green** vertices \( \Downarrow \) rejection proba does not increase.
Considering the modified oracles, define $E_{i+1} = \{ y \in D_{i+1} \mid y \text{ is red, i.e. } f_{i+1}(y) \neq \text{FOLD}[f_i, \alpha_i](y) \}$.

If no $\text{BAD}_i$ occurs, $\forall$ rejects with proba at most

$$\Pr_{s \in D_0} \left[ \exists i \in \{1, \ldots, r\}, s^2_i \in E_i \right] = \sum_{i=1}^{r} \Pr_{s \in D_0} \left[ s^2_i \in E_i \right] = \sum_{i=1}^{r} \frac{|E_i|}{|D_i|}.$$
Soundness analysis II

Considering the modified oracles, define $E_{i+1} = \{ y \in D_{i+1} \mid y \text{ is red, i.e. } f_{i+1}(y) \neq \text{FOLD}[f_i, \alpha_i](y) \}$.

If no $\text{BAD}_i$ occurs, $\nu$ rejects with proba at most

$$\Pr_{s \in D_0} \left[ \exists i \in \{1, \ldots, r\}, s^{2^i} \in E_i \right] = \sum_{i=1}^{r} \Pr_{s \in D_0} \left[ s^{2^i} \in E_i \right] = \sum_{i=1}^{r} \frac{|E_i|}{|D_i|}.$$  

Triangular inequality:

$$\Delta(\text{FOLD}[f_i, \alpha_i], f_{i+1}) + \Delta(f_{i+1}, \text{RS}_{i+1}) \geq \Delta(\text{FOLD}[f_i, \alpha_i], \text{RS}_{i+1}) \geq \delta_i.$$
Soundness analysis II

Considering the modified oracles, define $E_{i+1} = \{y \in D_{i+1} \mid y \text{ is red, i.e. } f_{i+1}(y) \neq \text{FOLD}[f_i, \alpha_i](y)\}$.

If no $\text{BAD}_i$ occurs, $\mathcal{V}$ rejects with proba at most

$$\Pr_{s \in D_0} \left[ \exists i \in \{1, \ldots, r\}, s^{2^i} \in E_i \right] = \sum_{i=1}^{r} \Pr_{s \in D_0} \left[ s^{2^i} \in E_i \right] = \sum_{i=1}^{r} \frac{|E_i|}{|D_i|}.$$ 

Triangular inequality:

$$\Delta(\text{FOLD}[f_i, \alpha_i], f_{i+1}) + \Delta(f_{i+1}, \text{RS}_{i+1}) \geq \Delta(\text{FOLD}[f_i, \alpha_i], \text{RS}_{i+1}) \geq \delta_i$$

We deduce that for all $i < r - 1$, $\frac{|E_{i+1}|}{|D_{i+1}|} \geq \delta_i - \delta_{i+1}$, since:

- If $\delta_{i+1} < \delta_i$, then $\delta_{i+1} < 1 - \sqrt{\rho}$, and thus $\delta_{i+1} = \Delta(f_{i+1}, \text{RS}_{i+1})$.
- Otherwise $\delta_i - \delta_{i+1} \leq 0$, thus it's clear.
Considering the modified oracles, define $E_{i+1} = \{ y \in D_{i+1} \mid y \text{ is red, i.e. } f_{i+1}(y) \neq \text{FOLD}[f_i, \alpha_i](y) \}$. 

If no $\text{BAD}_i$ occurs, $\mathcal{V}$ rejects with proba at most 

$$
\Pr_{s \in D_0} \left[ \exists i \in \{1, \ldots, r\}, s_0^i \in E_i \right] = \sum_{i=1}^r \Pr_{s \in D_0} \left[ s_0^i \in E_i \right] = \sum_{i=1}^r \frac{|E_i|}{|D_i|}.
$$

Triangular inequality:

$$
\Delta(\text{FOLD}[f_i, \alpha_i], f_{i+1}) + \Delta(f_{i+1}, RS_{i+1}) \geq \Delta(\text{FOLD}[f_i, \alpha_i], RS_{i+1}).
$$

We deduce that for all $i < r - 1$, $\frac{|E_{i+1}|}{|D_{i+1}|} \geq \delta_i - \delta_{i+1}$, since:

- If $\delta_{i+1} < \delta_i$, then $\delta_{i+1} < 1 - \sqrt{\rho}$, and thus $\delta_{i+1} = \Delta(f_{i+1}, RS_{i+1})$. 
  $\delta_i = \min \left( \Delta(f_i, RS_i), 1 - \sqrt{\rho} \right)$
- Otherwise $\delta_i - \delta_{i+1} \leq 0$, thus it’s clear.

Thus, \[
\sum_{i=1}^r \frac{|E_i|}{|D_i|} \geq \sum_{i=0}^{r-1} (\delta_i - \delta_{i+1}) \geq \delta_0 - \delta_r = \delta_0.
\]
Considering the modified oracles, define $E_{i+1} = \{y \in D_{i+1} \mid y \text{ is red, i.e. } f_{i+1}(y) \neq \text{FOLD}[f_i, \alpha_i](y)\}$.

If no BAD$_i$ occurs, $\mathcal{V}$ rejects with proba at most

$$\Pr_{s \in D_0} \left[ \exists i \in \{1, \ldots, r\}, s^{2^i} \in E_i \right] = \sum_{i=1}^r \Pr_{s \in D_0} \left[ s^{2^i} \in E_i \right] = \sum_{i=1}^r \frac{|E_i|}{|D_i|}.$$ 

Triangular inequality:

$$\Delta(\text{FOLD}[f_i, \alpha_i], f_{i+1}) + \Delta(f_{i+1}, \text{RS}_{i+1}) \geq \Delta(\text{FOLD}[f_i, \alpha_i], \text{RS}_{i+1}).$$

We deduce that for all $i < r - 1$, $\frac{|E_{i+1}|}{|D_{i+1}|} \geq \delta_i - \delta_{i+1}$, since:

- If $\delta_{i+1} < \delta_i$, then $\delta_{i+1} < 1 - \sqrt{\rho}$, and thus $\delta_{i+1} = \Delta(f_{i+1}, \text{RS}_{i+1})$. $\delta_i = \min \left( \Delta(f_i, \text{RS}_i), 1 - \sqrt{\rho} \right)$
- Otherwise $\delta_i - \delta_{i+1} \leq 0$, thus it’s clear.

Thus, $\sum_{i=1}^r \frac{|E_i|}{|D_i|} \geq \sum_{i=0}^{r-1} (\delta_i - \delta_{i+1}) \geq \delta_0 - \delta_r = \delta_0$.

Conclude: If $\delta = \Delta(f_0, \text{RS}_0) > 0$, then $\Pr(\mathcal{V} \text{ accept}) \leq O_\rho \left( \frac{n^2}{|F|} \right) + (1 - \min \left( \delta, 1 - \sqrt{\rho} \right))$. □
Conclusion

**In a nutshell:**

**Arithmetization:**

- valid execution trace $\rightsquigarrow$ RS codewords
- incorrect execution trace $\rightsquigarrow$ words far from any RS codewords

**Low-degree test:** key-ingredient is a “folding operator” such that:

- codeword of big RS code $\mapsto$ codeword of small RS code
- locally computable
- word far from big RS code $\mapsto$ words far from small RS code (w.h.p)

What about other codes?

Proximity testing to algebraic codes via IOPP, with linear prover and logarithmic verifier:

- Algebraic Geometry codes (joint work with Jade Nardi)
- Reed-Muller and Tensor Products of RS codes (joint work with Daniel)

The challenging part is to construct folding operators satisfying “distance preservation.”
In a nutshell:

**Arithmetization:**

- valid execution trace $\leadsto$ RS codewords
- incorrect execution trace $\leadsto$ words far from any RS codewords

**Low-degree test:** key-ingredient is a “folding operator” such that:

- codeword of big RS code $\mapsto$ codeword of small RS code
- locally computable
- word far from big RS code $\mapsto$ words far from small RS code (w.h.p)

**What about other codes?**

Proximity testing to algebraic codes via IOPP, with linear prover and logarithmic verifier:

- Algebraic Geometry codes (joint work with Jade Nardi)
  
  https://eccc.weizmann.ac.il/report/2020/165/

- Reed-Muller and Tensor Products of RS codes (joint work with Daniel)

The challenging part is to construct folding operators satisfying “distance preservation”.