Lecture 19
We have proved that arithmetic "circuit-like" computations have linear-size IOPs:

for every field \( \mathbb{F} \) of size \( \Omega(n) \) that is smooth \([\text{smoothness is for the LBT}]

\[
\operatorname{RICKS}(\mathbb{F}) \leq \operatorname{IOP} \left[ \begin{array}{l}
\varepsilon_c = 0, \varepsilon_s = 0.5, \Sigma = \mathbb{F}, \rho_t = O(n \log n), \nu t = O(n) \\
\lambda = 0(\log n), \eta = 0(\log n), \ell = O(n), \mu = 0(\log n)
\end{array} \right]
\]

The running time of the verifier is optimal, because just reading the statement takes \( \Omega(n) \) time. Similarly to before if we seek sublinear-time verification we need to consider problems whose description is smaller than computation size.

The holy grail would be a statement like the following:

\[
\operatorname{NTIME}(T) \leq \operatorname{IOP} \left[ \begin{array}{l}
\varepsilon_c = 0, \Sigma = \{0,1\}^T, \rho_t = O(T), \nu t = \text{poly}(n,\log T) \\
\varepsilon_s = 0.5, \lambda = O(T), \eta = \text{poly}(\log T)
\end{array} \right]
\]

This remains a challenging open question.

Instead, we will prove a "large alphabet" relaxation of the theorem:

**Theorem:** For every field \( \mathbb{F} \) of size \( \Omega(T) \) that is smooth \([\text{smoothness is for the LBT}]

\[
\operatorname{NTIME}(T, \mathbb{F}) \leq \operatorname{IOP} \left[ \begin{array}{l}
\varepsilon_c = 0, \Sigma = \mathbb{F}, \rho_t = O(T \log T), \nu t = \text{poly}(n,\log T) \\
\varepsilon_s = 0.5, \lambda = O(T), \eta = \text{poly}(\log T)
\end{array} \right]
\]
Informally, a machine is an automaton that can read/write to some type of memory. If memory = tapes then you get Turing machines. If memory = RAM then you get register machines (very close to how we think of a computer).

We are going to define languages that model machines that compute over finite fields. Let's start simple by first doing this for automata (i.e., no memory beyond internal state).

Consider:
- \( k \in \mathbb{N} \) - number of internal registers, i.e., a state is \( \mathbb{F}^k \)
- \( E : \mathbb{F}^k \rightarrow \mathbb{F}^k \) - transition function mapping current state to next state

A \( T \)-step computation looks as follows:

A computation requires \( O(1E1 + \log T) \) bits. The specified computation involves \( O(1E1 \cdot T) \) operations, exponentially more in \( T \). We are in fact interested in non-deterministic computations, and need an appropriate language.
Algebraic Automata

The bounded-halting problem for automata:

**def**: BH is the set of instances \((E, z, T)\) where \(E : F^k \to F^k, z \in F^n, T \in \mathbb{N}\) for which

**3** execution trace \(A_1, \ldots, A_k : \mathbb{N} \to F\) s.t.

**1** each step follows the transition function: \(\forall t \in \{0, 1, \ldots, T\-1\} E(A_1(t), \ldots, A_k(t)) = A_1(t+1), \ldots, A_k(t+1)\)

**2** the first \(n\) values of \(A_i\) are \(z\): \(A_i(0) = z\)

**3** the last value of \(A_1\) is 0: \(A_1(T) = 0\)

Let's massage this into a more convenient problem:

- Identify \([T]\) with a multiplicative subgroup \(H = \langle \omega \rangle \leq F\) s.t. \(|H| = T\).
  - Crucially, representing \(H\) requires only \(O(\log |H|)\) bits, rather than \(O(|H| \log |F|)\).
- We are interested to check not compute, so we translate the circuit \(E : F^k \to F^k\)
  into quadratic equations \(p_1, \ldots, p_m \in \mathbb{F}[x_1, \ldots, x_{k+1}]\) with \(m := O(|E|)\) and \(l := O(|E|)\) auxiliary vars.

\((E, z, T) \in \text{BH} \text{ iff } \exists \) augmented execution trace \(A_1, \ldots, A_k, B_1, \ldots, B_e : H \to F\)

- \(\forall t \in \{0, 1, \ldots, T-1\}\): \(\forall j \in \{m\} p_j(A_1(w^t), \ldots, A_k(w^t), A_1(w \cdot w^t), \ldots, A_k(w \cdot w^t), B_1(w^t), \ldots, B_e(w^t)) = 0\)
- \(A_i(0) = z, A_i(w^{T-1}) = 0\)

\(\Sigma \text{ size } (k+2lT = O(|E|))\)
Target-on-Subdomain Testing

Consider the setting where the verifier has oracle access to a function $f : L \rightarrow \mathbb{F}$ and wishes to check that $\hat{f}|_H \equiv \mathbf{z}$ for a given "target" function $\mathbf{z} : H \rightarrow \mathbb{F}$. (E.g. $\mathbf{z}$ is all 0's.)

We have seen this before: $\hat{f}(x)$ vanishes on $H$ if $\exists \hat{h}(x)$ s.t. $\hat{f}(x) - \mathbf{z}(x) = \hat{h}(x) \cdot v_H(x)$

Hence:

\[
P(L_{\mathbb{F} \cup H, \mathbf{z}}, f)
\] $\xrightarrow{\text{Compute } \hat{h}(x) = \frac{\hat{f}(x) - \hat{z}(x)}{v_H(x)}}$ $\xrightarrow{\text{f : L} \rightarrow \mathbb{F}}$ $\xrightarrow{\text{h : L} \rightarrow \mathbb{F}}$ $V((L_{\mathbb{F} \cup H, \mathbf{z}}))$
- Test that $h$ is $d$-close to $RS[F, L, d-1H]$
- Sample $y \in L$ and check $f(y) - \hat{z}(y) = h(y) \cdot v_H(y)$

**Completeness:** if $\hat{f}|_H \equiv \mathbf{z}$ then $h := \hat{h}|_L \in RS[F, L, d-1H]$ and passes check $\forall y \in L$

**Soundness:** if $\hat{f}|_H \not\equiv \mathbf{z}$ then $\forall h : L \rightarrow \mathbb{F}$ we have two cases:
- $h$ is $d$-far from $RS[F, L, d-1H] \rightarrow$ verifier accepts w.p. $\leq \epsilon_{\mathbb{F} \cup H}(s)$
- $h$ is $\delta$-close to $\hat{h}$ of degree $d-1H$ $\rightarrow$ $\hat{f}(x) - \hat{z}(x) \neq \hat{h}(x) \cdot v_H(x)$ so verifier accepts w.p. $\frac{d}{1H} + \delta$

Time complexity of the verifier:
- Ignore LDT because if using HFT $t_{\text{corr}} = O(\log |H|)$, which is small
- if $\mathbf{z} \not\equiv 0^H$ then: evaluate $v_H$ at $y$ and evaluate $\hat{z}$ at $y$ $\rightarrow \text{poly}(1H)$
- if $\mathbf{z} \equiv 0^H$ then: evaluate $v_H$ at $y$ $\rightarrow \text{poly}(1H)$ in general but $\text{poly}(\log |H|)$ if $H$ is a subgroup!

E.g. if $H$ is a multiplicative subgroup then $v_H(x) = x^{1H} - 1$. Crucial for us today.
IOP for Algebraic Automata

\[ PL(E, z, T, A) \]

- Run computation on trace \( A_1, \ldots, A_k \)
  augment it with \( B_1, \ldots, B_e \)
- For each \( i \in [k] \):
  compute \( f_i := \hat{A}_i|_L \in RS[\mathbb{F}, L, 1, H_1-1] \)
- For each \( i \in [k] \)
  compute \( g_i := \hat{B}_i|_L \in RS[\mathbb{F}, L, 1, H_1-1] \)
- For each \( j \in [m] \):
  compute \( h_j := \hat{h}_j(x)|_L \in RS[\mathbb{F}, L, 1, H_1-1] \)
  \( \hat{h}_j(x) := \frac{p_j(\hat{A}_1(x), \ldots, \hat{A}_k(x), \hat{B}_1(x), \ldots, \hat{B}_e(x))}{V_{H_1}(x)/(x-w^{T-1})} \)
- \( h_2 := \hat{h}_2(x)|_L \in RS[\mathbb{F}, L, 1, H_1-1] \)
  \( \hat{h}_2(x) := \frac{\hat{A}_1(x) - \hat{\lambda}(x)}{V_{H_1}(x)} \)
- \( h_0 := \hat{h}_0(x)|_L \in RS[\mathbb{F}, L, 1, H_1-1] \)
  \( \hat{h}_0(x) := \frac{\hat{A}(x)}{(x-w^{T-1})} \)

\[ V((E, z, T)) \]

\[ \{ f_i : L \rightarrow \mathbb{F}, i \in [k] \} \]
\[ \{ g_i : L \rightarrow \mathbb{F}, i \in [e] \} \]
\[ \{ h_j : L \rightarrow \mathbb{F}, j \in [m] \} \]
\[ h_2, h_0 : L \rightarrow \mathbb{F} \]

- Test each of the received function for the appropriate degree
  [we will come back to this]
- Sample \( T \in L \) and check that:
  \( \forall j \in [m] \)
  \( h_j(x) \frac{V_{H_1}(x)}{x-w^{T-1}} \neq p_j(f_1(x), \ldots, f_k(x), g_1(x), \ldots, g_e(x)) \)
  - \( h_2(x) V_{H_1}(x) = f_1(x) - \hat{\lambda}(x) \)
  - \( h_0(x) (x-w^{T-1}) = f_1(x) \)
Completeness

Suppose that $A_1, \ldots, A_k : [T] \to \mathbb{F}$ is a witness for $(E, z, T) \in \mathcal{BH}$.

- The prover can evaluate $E$ at each time step to augment the trace with $B_1, \ldots, B_k : [T] \to \mathbb{F}$ that satisfy all $m$ quadratic equations $p_1, \ldots, p_m$ derived from $E$. So the prover can find $\hat{h}_1(x), \ldots, \hat{h}_m(x)$.

- $A_i$ agrees with $z$ on first $n$ entries, and is 0 on last entry so $\hat{h}_i, \hat{h}_0$ too can be found.

Moreover:

- Proof length: $O((k+n+m)\log |L|) = O((k+n+m)\log |L|) = O(1E1\cdot T)$
- Query complexity: $O((k+n+m)\log |L|) = O(1E1\log T)$
- Prover time: $O((k+n+m)\log |L|) = O(1E1T\log T)$
- Verifier time: $O((k+n+m)\log |L|) + \text{poly}(n) = O(1E1\log T) + \text{poly}(n)$
Soundness

Suppose that \((E, z, T) \not\in BH\).

There are two cases:

1. One of the functions is far from RS.
   - \(\exists i \in [k] \) \(f_i\) is \(\varepsilon\)-far from \(RS[\mathbb{F}_L, IH] \)
   - \(\exists i \in [k] \) \(g_i\) is \(\varepsilon\)-far from \(RS[\mathbb{F}_L, IH] \)
   - \(\exists j \in [m] \) \(h_j\) is \(\varepsilon\)-far from \(RS[\mathbb{F}_L, IH] \)
   - \(h_0\) is \(\varepsilon\)-far from \(RS[\mathbb{F}_L, IH] \)

   \(\Rightarrow\) verifier accepts w.p. \(\leq \varepsilon \cdot \text{fail}(\varepsilon)\)

2. All functions are close to (unique) polynomials \(\{\hat{f}_i\}_{i \in [k]}, \{\hat{g}_i\}_{i \in [k]}, \{\hat{h}_j\}_{j \in [m]}, \hat{h}_0\) of the appropriate degree.
   - \(\exists j \in [m] \) \(\hat{h}_j(x) \cdot V_{h_0}(x) \neq \frac{p_3(f_l(w), -f_k(w)) h(x) \cdot \hat{g}_k(w)}{V_{h_0}(x)}\) consistency test passes w.p. \(\geq \frac{2 \cdot \text{H}_1 - 2 + (2k + b) \delta}{11}\)
   - \(\hat{h}_0(x) = \frac{\hat{h}_0(x)}{x - \omega^{-1}} \Rightarrow\) consistency check accepts w.p. \(\leq \frac{\text{H}_1 - 1}{11} + 2\delta\)
   - \(\hat{h}_0(x) = \frac{\hat{h}_0(x)}{x - \omega^{-1}} \Rightarrow\) consistency check accepts w.p. \(\leq \frac{\text{H}_1 - 1}{11} + 2\delta\)

Several options to make this \(\leq 1\):
- Set proximity parameter \(\varepsilon = O(\frac{1}{\text{H}_1})\)
- This requires setting repetition parameter \(t\) in FRI to \(t = O(1/\varepsilon)\) to ensure that \(\text{Err}(\varepsilon) = O(\varepsilon)\)
- Repeat consistency test \(t = O(\log(1/\varepsilon))\) times, as the term becomes \(\frac{2 \cdot \text{H}_1 - 2 + (2k + b) \delta}{t}\)
- Send random coefficients to prove & test \(\Sigma_i \delta f_i + \Sigma_i \delta g_i\) instead of individually
- Distortion statements imply the error becomes \(\frac{2 \cdot \text{H}_1 - 2 + 2\delta}{11}\) (due to column distance)
We now add memory:

If we extend the state with all of memory, we end up with $T^2$ variables — well beyond linear. **Observation:** it suffices to check correctness of memory operations, "what you work is what you read."

Consider the memory trace ordered first by address and then by time stamp:

<table>
<thead>
<tr>
<th>op</th>
<th>addr</th>
<th>time</th>
<th>val (read or written)</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>2</td>
<td>7</td>
<td>13</td>
</tr>
<tr>
<td>r</td>
<td>2</td>
<td>19</td>
<td>13</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>22</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>5</td>
<td>31</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>r</td>
<td>3</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td>2</td>
<td>9</td>
</tr>
</tbody>
</table>

The trace is correct iff for every two adjacent pairs $(op, addr, time, val), (op', addr', time', val')$ the following holds:

- if $addr = addr'$ then $time < time'$ and $(op = r \rightarrow val = val')$
- if $addr \neq addr'$ then $addr < addr'$

This leads to a language that represents machine computations...
Lemma: There is a polynomial-time reduction \( R \) s.t.

- \( R(E, \varepsilon, T) \) outputs quadratic equations \( p_1, \ldots, p_m \in \mathbb{F}[X_1, \ldots, X_{k+1}] \) with \( m, \ell = O(1E1) \)
- \( (E, \varepsilon, T) \in \text{BH} \) iff \( \exists \) augmented execution trace \( A_1, \ldots, A_k, B_1, \ldots, B_e : H \rightarrow \mathbb{F} \)

A permutation \( \pi : [T] \rightarrow [T] \) such that

- \( \forall t \in \{0, \ldots, T-1\} : \forall j \in [m] \ \ p_j \left( A_t(\omega^j), A_t(\omega^{2j}), \ldots, A_t(\omega^{j(T-1)}), B_t(\omega^j), \ldots, B_t(\omega^{j(T-1)}) \right) = 0 \)
- \( A_t \big|_{H_{in}} = 2, \ A_t(\omega^{T-1}) = 0 \)

Proof: Set \( p_1, \ldots, p_m \) to be the quadratic equations obtained by translating the transition function & also the logic for "what you write is what you read".

Completeness: Choose \( \pi \) to be the permutation that reorders the trace by address then time, so that the memory checks pass.

Soundness: for any choice of permutation \( \pi \), either some memory check fails, or the read/write operations are all correct so the transition function is fed the correct values.
Permutation Check

Consider the setting where the verifier has oracle access to \( f, g: L \rightarrow \mathbb{F} \) and wishes to check the claim:

\[
\exists \pi: H \rightarrow H \text{ s.t. } \forall a \in H \quad \hat{g}(a) = \hat{f}(\pi(a))
\]

**Idea:** the condition is equivalent to asking if \( \{\hat{g}(a)\}_{a \in H} \) and \( \{\hat{f}(\pi(a))\}_{a \in H} \) equal as multisets, which in turn is true if

\[
\prod_{a \in H} (x - \hat{g}(a)) \equiv \prod_{a \in H} (x - \hat{f}(\pi(a))).
\]

This directly leads to a protocol when \( H = \langle w \rangle \):

\[
P((L, H), (f, g))
\]

Compute partial products:

- \( \tilde{f}_\pi: L \rightarrow \mathbb{F} \text{ s.t. } \tilde{f}_\pi(w^\prime) = \prod_{j: s_i} (r - \hat{f}(w^\prime)) \)

- \( \tilde{g}_\pi: L \rightarrow \mathbb{F} \text{ s.t. } \tilde{g}_\pi(w^\prime) = \prod_{j: s_i} (r - \hat{g}(w^\prime)) \)

Compute \( h_1, h_2, h_3, h_4, h_5: L \rightarrow \mathbb{F} \) s.t.

\[
\begin{align*}
\hat{h}_1(x) &= \frac{f_\pi(x) - (r - f(x)) f_\pi(w^\prime x)}{v_H(x)/(x-1)} \\
\hat{h}_2(x) &= \frac{g_\pi(x) - (r - g(x)) g_\pi(w^\prime x)}{v_H(x)/(x-1)} \\
\hat{h}_3(x) &= \frac{f_\pi(x) - (r - f(x))}{(x-1)} \\
\hat{h}_4(x) &= \frac{g_\pi(x) - (r - g(x))}{(x-1)} \\
\hat{h}_5(x) &= \frac{f_\pi(x) - g_\pi(x)}{(x - w^\prime w^{-1})}
\end{align*}
\]

\[
V((L, H))
\]

Sample \( r \leftarrow \mathbb{F} \)

- Test that all received functions are LD.
- Sample \( r \leftarrow \mathbb{F} \) and check:

\[
\begin{align*}
h_1(x) &= v_H(x)/(x-1) \\
h_2(x) &= v_H(x)/(x-1) \\
h_3(x) &= v_H(x)/(x-1) \\
h_4(x) &= v_H(x)/(x-1) \\
h_5(x) &= v_H(x)/(x-1)
\end{align*}
\]