Lecture 3

Recap of last lecture

1. Reed-Solomon Fingerprinting.
   - Lets Alice and Bob determine whether their input vectors are equal, using communication that is logarithmic in the length of the vectors.
2. Freivalds' Protocol for Verifying Matrix Products.
   - Lets a verifier **check** that a matrix C equals the product of two matrices A and B.
   - Runtime of the verifier is linear in the size of the matrices.
   - Significantly faster than the best known algorithms for multiplying A and B).
3. Schwartz-Zippel lemma: Let $p \neq q$ be ℓ-variate polynomials of total degree at most d. Then $\Pr_{r \in \boldsymbol{F}^{\ell}}[p(r) = q(r)] \leq \frac{d}{|\boldsymbol{F}|}$.

Today

- Low-degree and multilinear extension polynomials.
- Our first interactive proof: the sum-check protocol.

Low-Degree and Multilinear Extensions

- Definition [**Extensions**]. Given a function $f\colon \{0,1\}^{\ell} \to \boldsymbol{F}$, a ℓ-variate polynomial g over **F** is said to **extend** f if $f(x) = g(x)$ for all $x \in \{0,1\}^{\ell}$.

- Definition [**Multilinear Extensions].** Any function $f\colon \{0,1\}^{\ell} \to \boldsymbol{F}$ has a **unique** multilinear extension (MLE), denoted $\tilde{f}$.

Low-Degree and Multilinear Extensions

- Definition [**Extensions**]. Given a function $f: \{0,1\}^{\ell} \rightarrow \boldsymbol{F}$, a ℓ-variate polynomial g over **F** is said to **extend** f if $f(x) = g(x)$ for all $x \in \{0,1\}^{\ell}$.

- Definition [**Multilinear Extensions].** Any function $f: \{0,1\}^{\ell} \rightarrow \boldsymbol{F}$ has a **unique** multilinear extension (MLE), denoted $\tilde{f}$.
  - Multilinear means the polynomial has degree at most 1 in each variable.
  - $(1 - x_1)(1 - x_2)$ is multilinear, $x_1^2 x_2$ is not.

$$f:\{0,1\}^2 \rightarrow \mathbf{F}$$

1	2
8	10

$$\tilde{f} : \mathbf{F}^2 \to \mathbf{F}$$

1	2	3	4	5	6
8	10	12	14	16	18
15	18	21	24	27	30
22	26	30	34	38	42
29	34	39	44	49	56
36	42	48	54	60	68

$$\tilde{f}(x_1, x_2) = (1 - x_1)(1 - x_2) + 2(1 - x_1)x_2 + 8x_1(1 - x_2) + 10x_1x_2$$

1	2	3	4	5	6
8	10	12	14	16	18
15	18	21	24	27	30
22	26	30	34	38	42
29	34	39	44	49	56
36	42	48	54	60	68

Can check:

$\tilde{f}(0,0) = 1$

$\tilde{f}(0,1) = 2$

$\tilde{f}(1,0) = 8$

$\tilde{f}(1,1) = 10$

Another (non-multilinear) extension of f:

$$g(x_1, x_2) = -x_1^2 + x_1 x_2 + 8\,x_1 + x_2 + 1$$

1	2	3	4	5	6
8	10	12	14	16	18
13	16	19	22	25	28
16	20	24	28	32	36
17	22	27	32	37	42
16	22	28	34	40	44

Can check:

$g(0,0) = 1$

$g(0,1) = 2$

$g(1,0) = 8$

$g(1,1) = 10$

Low-Degree and Multilinear Extensions

- Fact [VSBW13]: Given as input all 2^{ℓ} evaluations of a function $f: \{0,1\}^{\ell} \rightarrow \boldsymbol{F}$, for any point $r \in \boldsymbol{F}^{\ell}$ there is an $O(2^{\ell})$-time algorithm for evaluating $\tilde{f}(r)$.

- Note: If f is "structured", there may extensions g for which $g(r)$ can be evaluated **much** faster than $O(2^{\ell})$-time.

Low-Degree and Multilinear Extensions

- Fact [VSBW13]: Given as input all 2^ℓ evaluations of a function $f:\{0,1\}^\ell \rightarrow \boldsymbol{F}$, for any point $r \in \boldsymbol{F}^\ell$ there is an $O(2^\ell)$-time algorithm for evaluating $\tilde{f}(r)$.

- Note: If f is "structured", there may extensions g for which $g(r)$ can be evaluated **much** faster than $O(2^\ell)$-time.
  - We will see an example later when covering arithmetization of Boolean formulae.

The Sum-Check Protocol [LFKN90]

Sum-Check Protocol [LFKN90]

- Input: V given oracle access to a ℓ-variate polynomial g over field $\boldsymbol{F}$.
- Goal: compute the quantity:

$$\sum_{b_1 \in \{0,1\}} \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(b_1, \dots, b_\ell).$$

- **Start**: P sends claimed answer C_1. The protocol must check that:

$$C_1 = \sum_{b_1 \in \{0,1\}} \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(b_1, \dots, b_\ell).$$

- **Start**: P sends claimed answer C_1. The protocol must check that:

$$C_1 = \sum_{b_1 \in \{0,1\}} \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(b_1, \dots, b_\ell).$$

- **Round 1**: P sends **univariate** polynomial $s_1(X_1)$ claimed to equal:

$$\sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(X_1, b_2, \dots, b_\ell)$$

- **Start**: P sends claimed answer C_1. The protocol must check that:

$$C_1 = \sum_{b_1 \in \{0,1\}} \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(b_1, \dots, b_\ell).$$

- **Round 1**: P sends **univariate** polynomial $s_1(X_1)$ claimed to equal:

$$H_1(X_1) := \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(X_1, b_2, \dots, b_\ell)$$

- **Start**: P sends claimed answer C_1. The protocol must check that:

$$C_1 = \sum_{b_1 \in \{0,1\}} \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(b_1, \dots, b_\ell).$$

- **Round 1**: P sends **univariate** polynomial $s_1(X_1)$ claimed to equal:

$$H_1(X_1) := \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(X_1, b_2, \dots, b_\ell)$$

- V checks that $C_1 = s_1(0) + s_1(1)$.

- **Start**: P sends claimed answer C_1. The protocol must check that:

$$C_1 = \sum_{b_1 \in \{0,1\}} \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(b_1, \dots, b_\ell).$$

- **Round 1**: P sends **univariate** polynomial $s_1(X_1)$ claimed to equal:

$$H_1(X_1) := \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(X_1, b_2, \dots, b_\ell)$$

- V checks that $C_1 = s_1(0) + s_1(1)$.
- If this check passes, it is safe for V to believe that C_1 is the correct answer, so long as V believes that $s_1 = H_1$.
- How to check this? Just check that s_1 and H_1 agree at a random point r_1!

- **Start**: P sends claimed answer C_1. The protocol must check that:

$$C_1 = \sum_{b_1\in\{0,1\}} \sum_{b_2\in\{0,1\}} \dots \sum_{b_\ell\in\{0,1\}} g(b_1, \dots, b_\ell).$$

- **Round 1**: P sends **univariate** polynomial $s_1(X_1)$ claimed to equal:

$$H_1(X_1) := \sum_{b_2\in\{0,1\}} \dots \sum_{b_\ell\in\{0,1\}} g(X_1, b_2, \dots, b_\ell)$$

- V checks that $C_1 = s_1(0) + s_1(1)$.
- If this check passes, it is safe for V to believe that C_1 is the correct answer, so long as V believes that $s_1 = H_1$.
- How to check this? Just check that s_1 and H_1 agree at a random point r_1!
- V can compute $s_1(r_1)$ directly from P's first message, but not $H_1(r_1)$.

- **Start**: P sends claimed answer C_1. The protocol must check that:

$$C_1 = \sum_{b_1 \in \{0,1\}} \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(b_1, \dots, b_\ell).$$

- **Round 1**: P sends **univariate** polynomial $s_1(X_1)$ claimed to equal:

$$H_1(X_1) := \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(X_1, b_2, \dots, b_\ell)$$

- V checks that $C_1 = s_1(0) + s_1(1)$.
- V picks r_1 at random from $\boldsymbol{F}$ and sends r_1 to P.
- **Round 2**: They recursively check that $s_1(r_1) = H_1(r_1)$.

- **Start**: P sends claimed answer C_1. The protocol must check that:

$$C_1 = \sum_{b_1 \in \{0,1\}} \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(b_1, \dots, b_\ell).$$

- **Round 1**: P sends **univariate** polynomial $s_1(X_1)$ claimed to equal:

$$H_1(X_1) := \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(X_1, b_2, \dots, b_\ell)$$

- V checks that $C_1 = s_1(0) + s_1(1)$.
- V picks r_1 at random from $\boldsymbol{F}$ and sends r_1 to P.
- **Round 2**: They recursively check that $s_1(r_1) = H_1(r_1)$.

  i.e., that $s_1(r_1) = \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(r_1, b_2, \dots, b_\ell)$.

- **Start**: P sends claimed answer C_1. The protocol must check that:

$$C_1 = \sum_{b_1 \in \{0,1\}} \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(b_1, \dots, b_\ell).$$

- **Round 1**: P sends **univariate** polynomial $s_1(X_1)$ claimed to equal:

$$H_1(X_1) := \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(X_1, b_2, \dots, b_\ell)$$

- V checks that $C_1 = s_1(0) + s_1(1)$.
- V picks r_1 at random from $\boldsymbol{F}$ and sends r_1 to P.
- **Round 2**: They recursively check that $s_1(r_1) = H_1(r_1)$.

  i.e., that $s_1(r_1) = \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(r_1, b_2, \dots, b_\ell)$.

- **Round ℓ (Final round):** P sends univariate polynomial $s_\ell(X_\ell)$ claimed to equal

  $H_\ell := g(r_1, \dots, r_{\ell-1}, X_\ell)$.

- V checks that $s_{\ell-1}(r_{\ell-1}) = s_\ell(0) + s_\ell(1)$.
- V picks r_ℓ at random, and needs to check that $s_\ell(r_\ell) = g(r_1, \dots, r_\ell)$.
  - No need for more rounds. V can perform this check with one oracle query.

Analysis of the Sum-Check Protocol

Completeness and Soundness

- Completeness holds by design: If P sends the prescribed messages, then all of V's checks will pass.

Completeness and Soundness

- Completeness holds by design: If P sends the prescribed messages, then all of V's checks will pass.
- Soundness: If P does not send the prescribed messages, then V rejects with probability at least 1- $\frac{\ell \cdot d}{|F|}$, where d is the maximum degree of g in any variable.
- Proof is by induction on the number of variables ℓ.

Completeness and Soundness

- Completeness holds by design: If P sends the prescribed messages, then all of V's checks will pass.
- Soundness: If P does not send the prescribed messages, then V rejects with probability at least $1 - \frac{\ell \cdot d}{|\boldsymbol{F}|}$, where d is the maximum degree of g in any variable.
- Proof is by induction on the number of variables ℓ.
  - Base case: $\ell = 1$. In this case, P sends a single message $s_1(X_1)$ claimed to equal $g(X_1)$. V picks r_1 at random, checks that $s_1(r_1) = g(r_1)$.
  - By **Fact**, if $s_1 \neq g$, then $\Pr_{r_1 \in \boldsymbol{F}}[s_1(r_1) = g(r_1)] \leq \frac{d}{|\boldsymbol{F}|}$.

Soundness: Inductive Case

- Inductive case: $\ell > 1$.
  - Recall: P's first message $s_1(X_1)$ is claimed to equal
    $$H_1(X_1) := \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(X_1, b_2, \dots, b_\ell).$$
  - Then V picks a random r_1 and sends r_1 to P. They (recursively) invoke sum-check to confirm that $s_1(r_1) = H_1(r_1)$.

Soundness: Inductive Case

- Inductive case: $\ell > 1$.
  - Recall: P's first message $s_1(X_1)$ is claimed to equal
    $$H_1(X_1) := \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(X_1, b_2, \dots, b_\ell).$$
  - Then V picks a random r_1 and sends r_1 to P. They (recursively) invoke sum-check to confirm that $s_1(r_1) = H_1(r_1)$.
- By **Fact**, if $s_1 \neq H_1$, then $\Pr_{r_1 \in \boldsymbol{F}}[s_1(r_1) = H(r_1)] \leq \frac{d}{|\boldsymbol{F}|}$.

Soundness: Inductive Case

- Inductive case: $\ell > 1$.
  - Recall: P's first message $s_1(X_1)$ is claimed to equal
    $$H_1\,(X_1) := \sum_{b_2 \in \{0,1\}} \ldots \sum_{b_\ell \in \{0,1\}} g(X_1, b_2, \ldots, b_\ell).$$
  - Then V picks a random r_1 and sends r_1 to P. They (recursively) invoke sum-check to confirm that $s_1(r_1) = H_1(r_1)$.
- By **Fact**, if $s_1 \neq H_1$, then $\Pr_{r_1 \in \boldsymbol{F}}[s_1(r_1) = H(r_1)] \leq \frac{d}{|\boldsymbol{F}|}$.
- If $s_1(r_1) \neq H(r_1)$, P is left to prove a false claim in the recursive call.

Soundness: Inductive Case

- Inductive case: $\ell > 1$.
  - Recall: P's first message $s_1(X_1)$ is claimed to equal
    $$H_1\ (X_1) \coloneqq \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(X_1, b_2, \dots, b_\ell).$$
  - Then V picks a random r_1 and sends r_1 to P. They (recursively) invoke sum-check to confirm that $s_1(r_1) = H_1(r_1)$.
- By **Fact**, if $s_1 \neq H_1$, then $\Pr_{r_1 \in \boldsymbol{F}}[s_1(r_1) = H(r_1)] \leq \frac{d}{|\boldsymbol{F}|}$.
- If $s_1(r_1) \neq H(r_1)$, P is left to prove a false claim in the recursive call.
  - The recursive call applies sum-check to $g(r_1, X_2, \dots, X_\ell)$, which is ℓ-1 variate.
  - By induction, P fails to convince V in the recursive call with probability at least $1 - \frac{d(\ell-1)}{|\boldsymbol{F}|}$.

Soundness: Inductive Case

- Inductive case: $\ell > 1$.
  - Recall: P's first message $s_1(X_1)$ is claimed to equal
    $$H_1\ (X_1) \coloneqq \sum_{b_2 \in \{0,1\}} \dots \sum_{b_\ell \in \{0,1\}} g(X_1, b_2, \dots, b_\ell).$$
  - Then V picks a random r_1 and sends r_1 to P. They (recursively) invoke sum-check to confirm that $s_1(r_1) = H_1(r_1)$.
- By **Fact**, if $s_1 \neq H_1$, then $\Pr_{r_1 \in \boldsymbol{F}}[s_1(r_1) = H(r_1)] \leq \frac{d}{|\boldsymbol{F}|}$.
- If $s_1(r_1) \neq H(r_1)$, P is left to prove a false claim in the recursive call.
  - The recursive call applies sum-check to $g(r_1, X_2, \dots, X_\ell)$, which is ℓ-1 variate.
  - By induction, P fails to convince V in the recursive call with probability at least $1 - \frac{d(\ell-1)}{|\boldsymbol{F}|}$.
- **Summary:** if $s_1 \neq H_1$, the probability V accepts is at most:
  $$\Pr_{r_1 \in \boldsymbol{F}}[s_1(r_1) = H(r_1)] + \Pr_{r_2, \dots, r_\ell \in \boldsymbol{F}}[\text{V accepts} | s_1(r_1) \neq H(r_1)]$$
  $$\leq \frac{d}{|\boldsymbol{F}|} + \frac{d(\ell-1)}{|\boldsymbol{F}|} \leq \frac{d\ell}{|\boldsymbol{F}|}.$$

Costs of the Sum-Check Protocol

- Total communication is $O(d\ell)$ field elements.
  - P sends ℓ messages, each a univariate polynomial of degree at most d. V sends $\ell - 1$ messages, each consisting of one field elements.

Costs of the Sum-Check Protocol

- Total communication is $O(d\ell)$ field elements.
  - P sends ℓ messages, each a univariate polynomial of degree at most d. V sends $\ell - 1$ messages, each consisting of one field elements.
- V's runtime is:

$O(d\ell + [\textit{time required to evaluate } g \textit{ at one point}])$.

Costs of the Sum-Check Protocol

- Total communication is $O(d\ell)$ field elements.
  - P sends ℓ messages, each a univariate polynomial of degree at most d. V sends $\ell - 1$ messages, each consisting of one field elements.
- V's runtime is:

$$O(d\ell + [\textit{time required to evaluate } g \textit{ at one point}]).$$

- P's runtime is at most:

$$O\left(d \cdot 2^{\ell} \cdot [\textit{time required to evaluate } g \textit{ at one point}]\right).$$