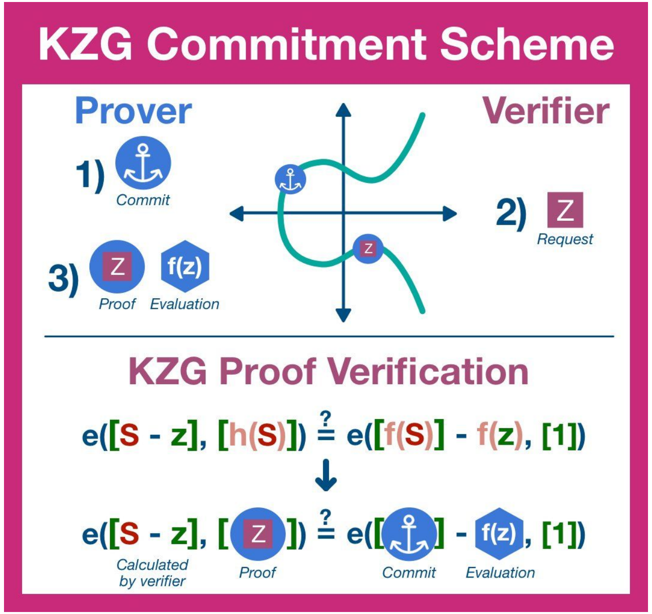
FRI

23.03.23 ZK-School

Kate commitment (by hand)



FRI

Fast Reed-Solomon Interactive Oracle Proof of Proximity

- Fast
- Read-Solomon
- Interactive Oracle Proof
- Proximity

FRI 이해하기

1. FRI 알고리즘: Verifier가 뭘 확인하면 True라고?
2. FFT: 다항식을 점으로 만들면 뭐가 좋은데?
3. Equation 만들기: Verifier가 확인하는 f(x)에 대한 다항식을 어떻게 만든거야?
4. Degree 증명: 랜덤 값 r에서 방정식이 만족하면 모든 다항식이 유효하다고 할 수 있다고?
5. Starkware의 FRI: FRI는 이해했어, 그럼 starkware에서는 이걸 그대로 사용해서 증명해?

1. Verifier 가 뭘 확인하면 True 라고?

- Prover는 다항식 f(x)를 보내는 게 아니라, 모든 x 값에 대한 f(x)의 결과값 (a, f(a))을 보낸다.
- commitment를 Prover가 Verifier에게 모두 전송하는 것은 아니고, 모든 값을 미리 구해서 Oracle에 퍼블리시 해놓는다.
- Verifier는 임의의 점 r를 선택하여 f(r) 값을 oracle에 요청해서 그 상수 값을 받는다.
- Verifier는 받아온 상수값으로 **2개의 방정식**이 만족하는지 확인하고, 모두 만족한다면 True라고 이야기한다.

1. Verifier 가 뭘 확인하면 True 라고?

Prover

Verifier

- Verifier는 받아온 상수값으로 **2개의 방정식**이 만족하는지 확인하고, 모두 만족한다면 True라고 이야기한다.

$$f(r\omega^2) - f(r\omega) - f(r) = \frac{Z_H(r)}{(r-\omega^{n-2})(r-\omega^{n-1})} * g(r)$$

$$f(r) - B(r) = D(r) * g'(r)$$

2. FFT 다항식을 점으로 만들면 뭐가 좋은데?

Prover는 Oracle에 다항식의 결과값의 집합들을 퍼블리시 해놓으면 된다.

$$F(\omega) = \{f(\omega^0), f(\omega^1), \dots, f(\omega^n)\}$$

$$f(x) = a_0 + a_1 x + \dots + a_n x^n \rightarrow f(\omega^0), f(\omega^1), \dots, f(\omega^n)$$

$$f(\omega^0) = a_0 + a_1\omega^0 + \dots + a_n\omega^0$$

$$f(\omega^1) = a_0 + a_1\omega^1 + \dots + a_n\omega^1$$

...

$$f(\omega^n) = a_0 + a_1\omega^n + \dots + a_n\omega^n$$

2. FFT 다항식을 점으로 만들면 뭐가 좋은데?

f(w^2)를 미리 계산을 해놓은 후, f(w)을 계산할 때 미리 계산했던 f(w^2)와 공통된 형태를 활용하는 것이다.

$$f(\omega) = f_L(\omega^2) + \omega f_R(\omega^2)$$

FFT는 다항식 f(x)를 특정한 x값에 대한 y값을 계산하여 점의 형태로 바꾸는 것을 말한다.

- n_th root of unity

$$\omega^i \in \{x | x^n = 1\}$$

2. FFT 다항식을 점으로 만들면 뭐가 좋은데?

$$f(x) = a_0 + a_1x + ... + a_nx^n$$

$$f(x) = (a_0 + a_2x^2 + a_4x^4 + ...) + x(a_1 + a_3x^2 + a_5x^4 + ...)$$

$$f_L(x) = a_0 + a_2x^2 + a_4x^4 + ...$$
$$f_R(x) = a_1 + a_3x^2 + a_5x^4 + ...$$

$$\Longrightarrow \quad f(x) = f_L(y) + xf_R(y)$$

$$f_L(\omega^0) = a_0 + a_2\omega^{0^2} + a_4\omega^0 + ... + a_ny^{\frac{n}{2}}$$
$$f_L(\omega^2) = a_0 + a_2\omega^2 + a_4\omega^4 + ... + a_n\omega^{\frac{n}{2}}$$

$$\Longrightarrow \quad f_L(y) = f_{L1}(y') + yf_{R1}(y')$$

3. Equation : Verifier가 f(x)에 대한 다항식 어떻게 만든거야?

Prover 가 증명 하는 것 : 함수 f는 피보나치 수열을 연산하는 함수이며, f의 마지막 값을 p로 나눈 나머지는 b이다.

1. $f(\omega^{i+2}) = f(\omega^i) + f(\omega^{i+1})$
2. $f(1) = 1, f(\omega) = 1, f(\omega^{n-1}) = b$

3. Equation : Verifier가 f(x)에 대한 다항식 어떻게 만든거야?

1번 방정식 : 1. $f(\omega^{i+2}) = f(\omega^i) + f(\omega^{i+1})$

$$f(\omega^{i+2}) - f(\omega^{i+1}) - f(\omega^i) = 0$$

$$f(\omega^2 * \omega^i) - f(\omega * \omega^i) - f(\omega^i) = 0$$

$$Z_H(x) = (x-1) * (x-\omega) * \ldots * (x-\omega^{n-1})$$

$$f(\omega^2 x) - f(\omega x) - f(x) = Z_H(x) * g(x)$$

3. Equation : Verifier가 f(x)에 대한 다항식 어떻게 만든거야?

1번 방정식 : 1. $f(\omega^{i+2}) = f(\omega^{i}) + f(\omega^{i+1})$

$$x = \omega^{n-3} \Rightarrow f(\omega^{n-1}) - f(\omega^{n-2}) - f(\omega^{n-3}) = 0$$

$$x = \omega^{n-2} \Rightarrow f(\omega^{n}) - f(\omega^{n-1}) - f(\omega^{n-2}) = 0$$

$$x = \omega^{n-1} \Rightarrow f(\omega^{n+1}) - f(\omega^{n}) - f(\omega^{n-1}) = 0$$

$$x \in H = \{1, \omega, ..., \omega^{n-1}\} - \{\omega^{n-2}, \omega^{n-1}\}$$

$$f(\omega^2 x) - f(\omega x) - f(x) = \frac{Z_H(x)}{(x - \omega^{n-2})(x - \omega^{n-1})} * g(x)$$

3. Equation : Verifier가 f(x)에 대한 다항식 어떻게 만든거야?

2번 방정식 : 2. $f(1) = 1, f(\omega) = 1, f(\omega^{n-1}) = b$

$$B(x) = \frac{(x-\omega)(x-\omega^{n-1})}{(1-\omega)(1-\omega^{n-1})} + \frac{(x-1)(x-\omega^{n-1})}{(\omega-1)(\omega-\omega^{n-1})} + b * \frac{(x-1)(x-\omega)}{(\omega^{n-1}-1)(\omega^{n-1}-\omega)}$$

$$D(x) = (x-1)(x-\omega)(x-\omega^{n-1})$$

$$f(x) - B(x) = D(x) * g'(x)$$

4. Degree 증명 : 랜덤 값 r에서 방정식이 만족하면 모든

다항식이 유효하다고?

5. Starkware의 FRI

- Split to even and odd powers

$$P_0(x) = g(x^2) + xh(x^2)$$

- Get a random β

- Consider the new function:
  $$P_1(y) = g(y) + \beta h(y)$$

- Example:
  $$P_0(x) = 5x^5+3x^4+7x^3+2x^2+x+3$$

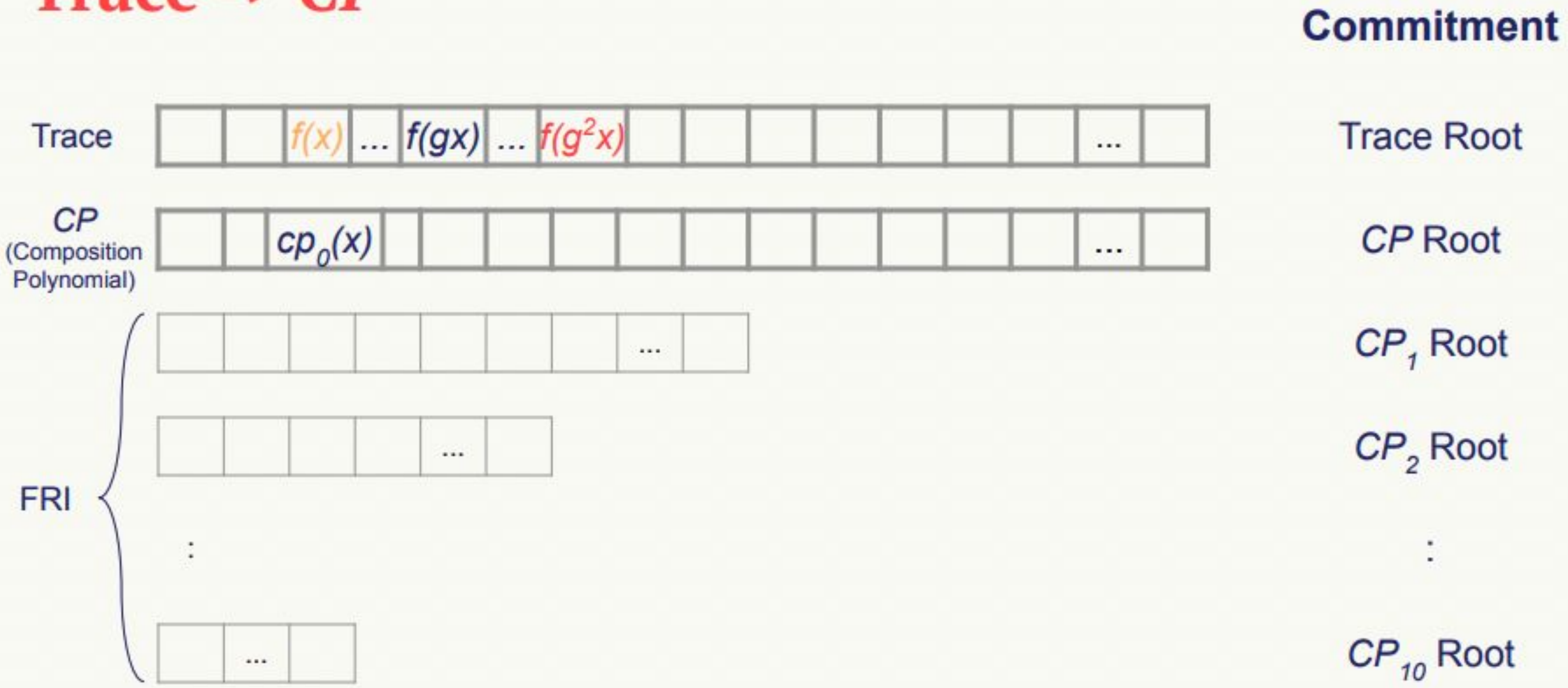
  $g(y)$ $\quad 3y^2 \quad 2y \quad 3$

  $h(y)$ $\quad 5y^2 \quad 7y \quad 1$

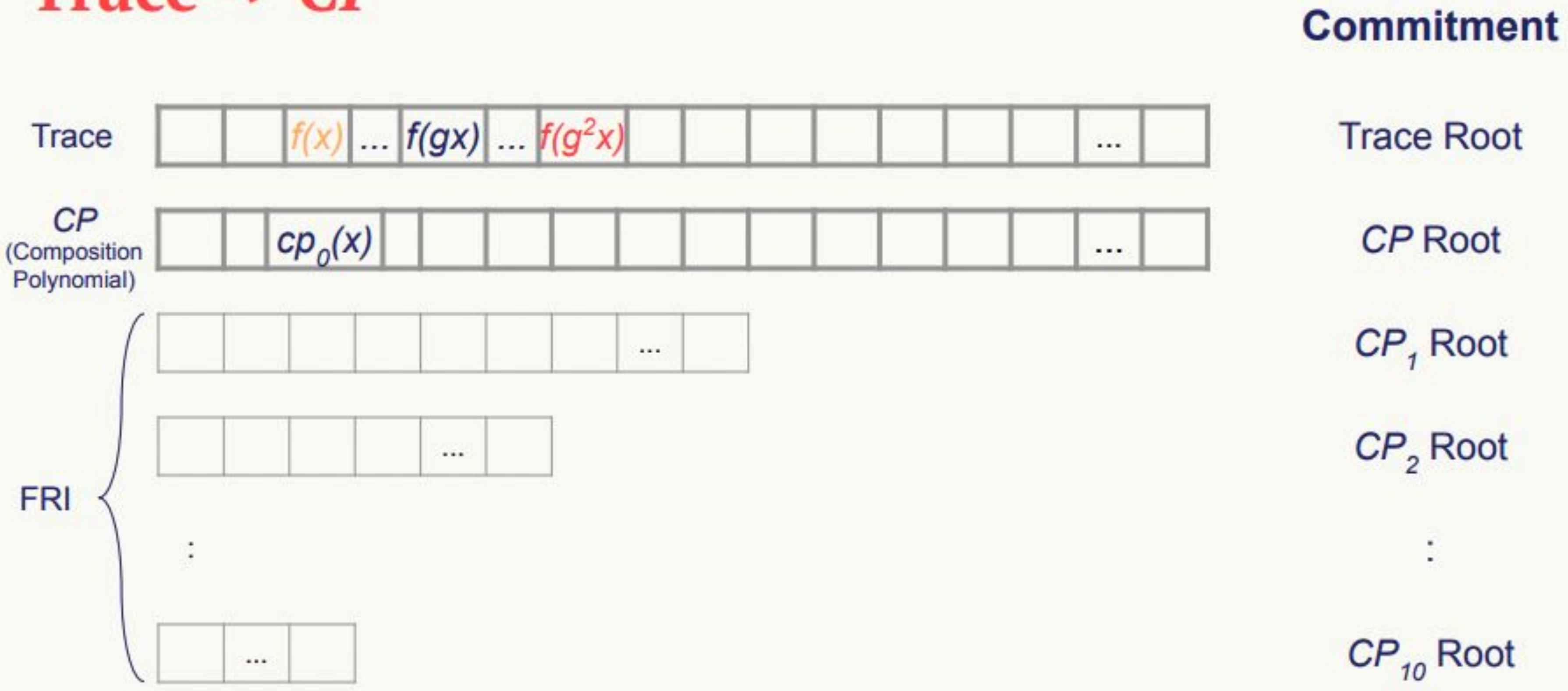
- $$P_1(y) = 3y^2+2y+3+\beta(5y^2+7y+1)$$
  $$= (3+5\beta)y^2+(2+7\beta)y+3+\beta$$

STARKWARE STARK 101

Trace -> *CP*


Commitment
Trace
f(x)
...
f(gx)
...
f(g²x)
...
Trace Root
CP
(Composition Polynomial)
cp₀(x)
...
CP Root
FRI
...
CP₁ Root
...
CP₂ Root
:
:
...
CP₁₀ Root

Trace -> *CP*


Commitment
Trace
f(x)
...
f(gx)
...
f(g²x)
...
Trace Root
CP
(Composition
Polynomial)
cp₀(x)
...
CP Root
FRI
...
CP₁ Root
...
CP₂ Root
:
:
...
CP₁₀ Root

Decommitment Phase (for query x)