0.1 Preliminaries

Before delving into Plonk, let's first consider a few useful proving system gadgets.

Let Ω be some multiplicative cyclic subgroup of $\mathbb{F}_p$ of size k.

Let $f \in \mathbb{F}_p^{(\leq d)}[X] \qquad (d \geq k)$ $\boxed{\text{Verifier has } f}$

Now we need to construct efficient Poly-IOPs for two tasks described in below sections.

0.1.1 ZeroTest

Prove that f is identically zero on Ω.

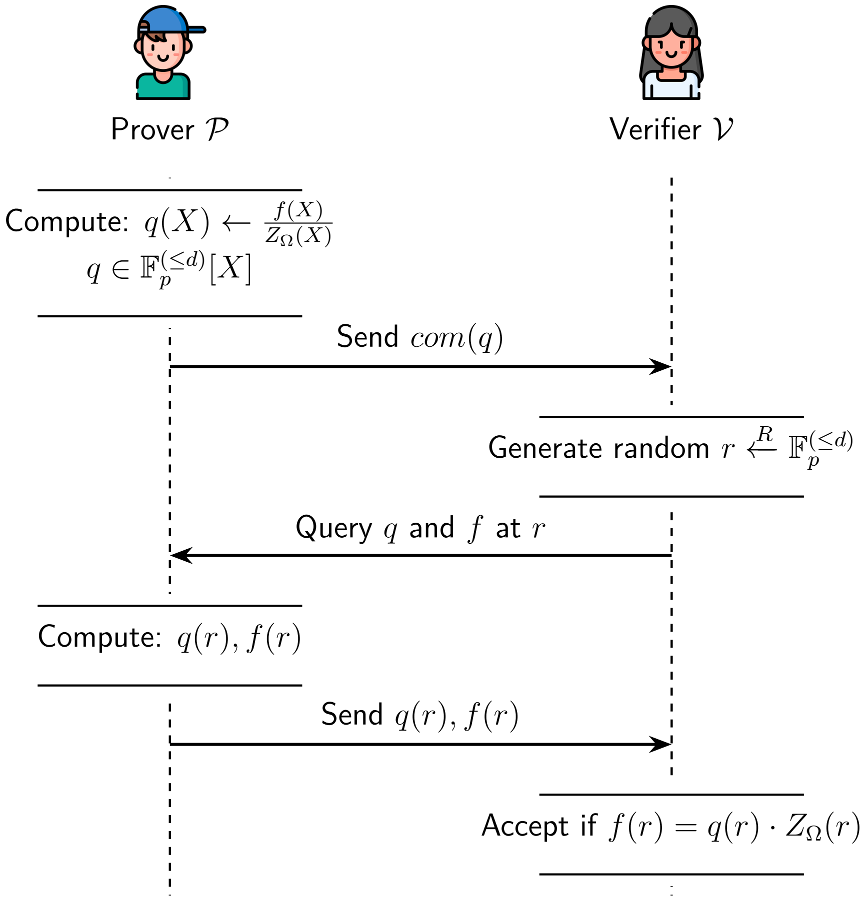

Figure 0.1: Protocol between prover $\mathcal{P}$ and verifier $\mathcal{V}$ for ZeroTest.

Lemma 0.1. f is zero on Ω if and only if $f(X)$ is divisible by $Z_\Omega(X)$.

Remark. This protocol is complete and sound, assuming d/p is negligible due to Schwartz-Zippel Lemma.

0.1.2 Prescribed Permutation Check

Definition 0.2. $W : \Omega \to \Omega$ is a permutation of Ω if $\forall i \in [k], W(\omega^i) = \omega^j$ is a bijection.

Example. Example $(k = 3): \quad W(\omega^0) = \omega^2, \quad W(\omega^1) = \omega^0, \quad W(\omega^2) = \omega^1$

Let f and g be polynomials in $\mathbb{F}_p^{(\leq d)}[X]$. Verifier has commitments $com(f)$, $com(g)$, $com(W)$.

Goal: prover wants to prove that $f(y) = g(W(y))$ for all $y \in \Omega$, equivalent to $g(\Omega)$ is the same as $f(\Omega)$, permuted by the prescribed W.

Naive way of doing this is by running **ZeroTest** on $f(y) - g(W(y)) = 0$ on Ω, however that would result in proving needing to manipulate polynomials of degree k^2, resulting in quadratic prover time. We can reduce this to a product-check on a polynomial of degree $2k$.

Remark. Observation: If $(W(\alpha), f(\alpha))_{\alpha\in\Omega}$ is a permutation of $(\alpha, g(\alpha))_{\alpha\in\Omega}$ then $f(y) = g(W(y))$ for all $y \in \Omega$.

Example.

$$\text{Proof by example: } W(\omega^0) = \omega^2, \quad W(\omega^1) = \omega^0, \quad W(\omega^2) = \omega^1$$

$$\text{Right tuple: } (\omega^0, g(\omega^0)), (\omega^1, g(\omega^1)), (\omega^2, g(\omega^2))$$

$$\text{Left tuple: } (\omega^2, f(\omega^0)), (\omega^0, f(\omega^1)), (\omega^1, f(\omega^2))$$

You can see that the tuple on the right is formed by $(\alpha, g(\alpha))_{\alpha\in\Omega}$ and in the tuple on the left the first part is a permuted by W image of α. Meaning, that if, for example, ω_0 is mapped to ω_2, then the only way $(\omega^2, g(\omega^2)) = (\omega^2, f(\omega^0))$ is if $g(\omega^2) = f(\omega^0)$. And the same thing holds for all other pairs, resulting in requirement for $f(y) = g(W(y))$ for all $y \in \Omega$.

Lemma 0.3. Let:

1. $\hat{f}(X,Y) = \prod_{\alpha\in\Omega}(X - Y \cdot W(\alpha) - f(\alpha))$
2. $\hat{g}(X,Y) = \prod_{a\in\Omega}(X - Y \cdot a - g(a))$

- (bivariate polynomials).

$\hat{f}(X,Y) = \hat{g}(X,Y) \Leftrightarrow (W(\alpha), f(\alpha))_{\alpha\in\Omega}$ is a permutation of $(\alpha, g(\alpha))_{\alpha\in\Omega}$.

To prove, use the fact that $\mathbb{F}_p[X,Y]$ is a unique factorization domain: $\hat{f}$ and $\hat{g}$ factor uniquely, so if these are identical, their prime factors are identical, for which the lemma falls very easily.

The complete protocol.

1. Verifier generates random $r, g \leftarrow \mathbb{F}_p^{(\leq d)}$
2. Run **ProductCheck** on $\hat{f}(r,s) = \hat{g}(r,s) : \prod_{a\in\Omega}\left(\frac{r - s\cdot W(a) - f(a)}{r - s\cdot a - g(a)}\right) = 1$

This would imply that $\hat{f}(X,Y) = \hat{g}(X,Y)$ with high probability due to Schwartz-Zippel Lemma.

Remark. Complete and sound, assuming $2d/p$ is negligible.

0.2 Plonk Arithmetization

Assume that we have a certain arithmetic circuit $\mathtt{C}$ with a $|\mathtt{C}|$ number of gates and $|\mathcal{I}| = |\mathcal{I}_x| + |\mathcal{I}_w|$ number of inputs. We encode this circuit, recording its computation trace in a table, where rows represent the state per each gate, while columns are of the form (a, b, c), where a and b are left and right inputs, and c is the output of the gate. In this manner, the output of the last gate corresponds to the output of the circuit.

Example. Consider this circuit: $(x_1 + x_2) \times (x_1 + w_1)$. Suppose we set $x_1 = 5, x_2 = 6, w_1 = 1$. Then, the computation trace would be following:

inputs:	5,	6,	1
Gate 0:	5,	6,	11
Gate 1:	6,	1,	7
Gate 2:	11,	7,	77

0.2.1 Encoding the trace as a polynomial

Let $d = 3|C| + |\mathcal{I}|$ and $\Omega = \{1, \omega^1, \omega^2, \ldots, \omega^{d-1}\}$. Then we would like to interpolate a polynomial $T \in \mathbb{F}_p^{(\leq d)}[X]$ to encode the entire computation trace in a succinct, processing-prone form.

1. T **encodes all inputs:** $T(\omega^{-j}) =$ input $\#j$ for $j = 1, \ldots, |\mathcal{I}|$
2. T **encodes all wires:** $\forall \ell = 0, \ldots, |\mathtt{C}| - 1:$
   - $T(\omega^{3\ell})$: left input to gate $\#\ell$
   - $T(\omega^{3\ell+1})$: right input to gate $\#\ell$
   - $T(\omega^{3\ell+2})$: output of gate $\#\ell$

Remark. In this way, we obtain the polynomial T in evaluation form. It is possible to compute coefficients of T using FFT in $O(d \log(d))$. More on that in the subsequent lectures.

Example. For our example circuit, we have $|\mathtt{C}| = 3$ and $|\mathcal{I}| = 3$, therefore $\deg(T) = 11$. The prover interpolates the polynomial $T(X)$ such that:

inputs:	$T(\omega^{-1}) = 5$,	$T(\omega^{-2}) = 6$,	$T(\omega^{-3}) = 1$,
gate 0:	$T(\omega^{0}) = 5$,	$T(\omega^{1}) = 6$,	$T(\omega^{2}) = 11$,
gate 1:	$T(\omega^{3}) = 6$,	$T(\omega^{4}) = 1$,	$T(\omega^{5}) = 7$,
gate 2:	$T(\omega^{6}) = 11$,	$T(\omega^{7}) = 7$,	$T(\omega^{8}) = 77$

0.3 Proving the validity of T

After prover $\mathcal{P}(pp, x, w)$ has constructed T, it commits it and sends to the verifier $\mathcal{V}(vp, x)$. Latter must verify (meaning former must prove) four points about the validity of constructed T, which we will describe in next sections.

0.3.1 Trace Polynomial encodes the correct inputs

Both prover and verifier interpolate a polynomial $\nu(X) \in \mathbb{F}_p^{(\leq d)}[X]$ that encodes the x-th input to the $\mathtt{C}$:

$$\text{for } j = 1, \ldots, |\mathcal{I}_x| : \nu(\bar{\omega}^j) = \text{input } \#j$$

Remark. Constructing $\nu(X)$ is linear in $|x|$ $(O_\lambda(|x|))$.

Let $\Omega_{\mathsf{inp}} := \{\omega^{-1}, \omega^{-2}, ..., \omega^{-|\mathcal{I}_x|}\} \subseteq \Omega$, then proving polynomial T encoding is done with *ZeroTest* on Ω_{inp}:

$$T(y) - \nu(y) = 0 \quad \forall y \in \Omega_{\mathsf{inp}}$$

> **Example.** In our example, $\nu(\omega^{-1}) = 5$, $\nu(\omega^{-2}) = 6$.

0.3.2 Every gate is evaluated correctly

Encode gate types using a *selector* polynomial $S(X)$: $S(X) \in \mathbb{F}_p^{\leq d}[X]$ such that $\forall\, \ell = 0, ..., |\mathtt{C}| - 1$:

- $S(\omega^{3l}) = 1$ if gate $\#\ell$ is an addition gate
- $S(\omega^{3l}) = 0$ if gate $\#\ell$ is a multiplication gate

Then, $\forall\, y \in \Omega_{\text{gates}} := \{1, \omega^3, \omega^6, \omega^9, ..., \omega^{3(|C|-1)}\}$:

$$S(y) \cdot [T(y) + T(\omega y)] + (1 - S(y)) \cdot T(y) \cdot T(\omega y) = T(\omega^2 y)$$

where $T(y)$ and $T(\omega y)$ are left and right inputs correspondingly.

This means, that once again we can narrow our check down to *ZeroTest* for $\forall\, y \in \Omega_{\text{gates}}$:

$$S(y) \cdot [T(y) + T(\omega y)] + (1 - S(y)) \cdot T(y) \cdot T(\omega y) - T(\omega^2 y) = 0$$

> **Example.** That is, in our $\mathtt{C}$, since gates 0 and 1 are addition and 2 gate is multiplication, we have the following encoding for selector polynomial:
>
> | inputs: | 5, | 6, | 1 | $S(X)$ |
> |---|---|---|---|---|
> | Gate 0 (ω^0): | 5, | 6, | 11 | 1 |
> | Gate 1 (ω^3): | 6, | 1, | 7 | 1 |
> | Gate 2 (ω^6): | 11, | 7, | 77 | 0 |
>
> You can see by substituting $S(y)$ with actual values from the table, that if a selector polynomial evaluates as 1, then multiplication part of the proved constraint is leveling out - vice versa for 0.

0.3.3 The wiring is implemented correctly

In this part, we need to encode the wires of $\mathtt{C}$ to prove connection of inputs and outputs in gates.

> **Example.** For our table:
>
> | ω^{-1}, ω^{-2}, ω^{-3}: | 5, | 6, | 1 |
> |---|---|---|---|
> | 0: ω^0, ω^1, ω^2: | 5, | 6, | 11 |
> | 1: ω^3, ω^4, ω^5: | 6, | 1, | 7 |
> | 2: ω^6, ω^7, ω^8: | 11, | 7, | 77 |
>
> We need following constraints:
>
> $$T(\omega^{-2}) = T(\omega^1) = T(\omega^3) \tag{1}$$
> $$T(\omega^{-1}) = T(\omega^0) \tag{2}$$
> $$T(\omega^2) = T(\omega^6) \tag{3}$$
> $$T(\omega^{-3}) = T(\omega^4) \tag{4}$$
>
> For that matter, define a polynomial $W = \Omega \rightarrow \Omega$ that implements a rotation:
>
> $$W(\omega^{-2}, \omega^1, \omega^3) = (\omega^1, \omega^3, \omega^{-2}), \quad W(\omega^{-1}, \omega^0) = (\omega^0, \omega^{-1}), \quad \ldots$$

Lemma 0.4. Lemma: $\forall y \in \Omega : T(y) = T(W(y)) \implies$ wire constraints are satisfied. This may be proven using *prescribed permutation check*.

0.3.4 Output of the last gate conforms with expected

Let α be the expected output of $\mathtt{C}$. Then, apply *ZeroTest* on:

$$T(\omega^{3|\mathtt{C}|-1}) - \alpha = 0$$

0.4 Setup procedure

Not accounting for the details of selected poly-commit scheme (i.e KZG, FRI, etc.) the setup procedure preprocesses the circuit C and outputs for the prover selector and wiring polynomials S and W, and for the verifier respective commitments $com(S)$ and $com(W)$.

Remark. This setup procedure is untrusted.

0.5 Summary

Plonk

Arithmetic circuit $\mathtt{C}$ with standard addition and multiplication gates with fan-in 2.

Review

- ✓ $T \in \mathbb{F}_p^{(\leq d)}[X]$ - Computation trace encoding polynomial.
- ✓ $S \in \mathbb{F}_p^{(\leq d)}[X]$ - Selector polynomial encoding the gates.
- ✓ $W = \Omega \to \Omega$ - Wiring polynomial connecting values in the computation trace table.

Setup

- ✓ $\mathcal{P}(pp, x, w) \leftarrow S, W$
- ✓ $\mathcal{V}(vp, x) \leftarrow com(S), com(W)$

$\mathcal{P}(\mathsf{pp}, \mathsf{x}, \mathsf{w}) \rightleftarrows \mathcal{V}(\mathsf{vp}, \mathsf{x})$

1. Gates: $S(y) \cdot [T(y) + T(\omega y)] + (1 - S(y)) \cdot T(y) \cdot T(\omega y) - T(\omega^2 y) = 0 \qquad \forall y \in \Omega_{\text{gates}}$
2. Inputs: $T(y) - v(y) = 0 \qquad \forall y \in \Omega_{\text{inp}}$
3. Wires: $T(y) - T(W(y)) = 0$ (using prescribed perm. check) $\qquad \forall y \in \Omega$
4. Output: $T(\omega^{3|C|-1}) = 0$ (output of last gate $= 0$)