KZG Polynomial Commitment Study Notes

LI CHENCHEN

1 Introduction

Commitment schemes are fundamental components of many cryptographic protocols. A secure commitment scheme allows a committer to publish a value, called the commitment, which binds committer to a message (binding) without revealing it (hiding). Later, committer may open the commitment and reveal the committed message to a verifier.
Kate-Zaverucha-Goldberg (KZG) polynomial commitment is a powerful tool in blockchain applications. It allows prover to compute a commitment to a polynomial and prover may show that the value of the polynomial at certain position is equal to a claimed value. The use case of KZG polynomial commitment including "witness generation" of L2 computation for zk-rollups (Scoll specifically use KZG for its polynomial commitment) and for upcoming ethereum proto-danksharding in Dencun upgrade, ethereum plans to use KZG commitment to generate commitment of data blob for execution layer.

2 Preliminaries

To understand how KZG polynomial commitment works, it is suggested to have some knowledge about abstract algebra, elliptic curve pairings, polynomials and some crptography assumptions. I will give brief introduction to them.

2.1 Abstract Algebra

2.1.1 Cyclic Group

For any prime number p, multiplicative group of order p is defined to be

$$\mathbb{Z}_p^* = \{a \mid 1 \leq n < p\}$$

$\mathbb{Z}_p^*$ is a finite abelian group in which for all elements a in $\mathbb{Z}_p^*$, $\gcd(a, p) = 1$. Hence by Euler's Theorem, we have

$$a^{\phi(p)} = 1$$

where $\phi(p) = p - 1$ for all prime number p.
Therefore, every element of $\mathbb{Z}_p^*$ is a generator of $\mathbb{Z}_p^*$, i.e.

$$\forall g \in \mathbb{Z}_p^*, \text{we have } \{g, g^2, \ldots, g^{p-1}\} = \mathbb{Z}_p^*$$

2.1.2 Finite Field and Field Extension

The characteristic of any finite field is a prime number and , the cardinality of a field with characteristic p must be p^n, where p is some prime number and $n \in \mathbb{N}$. The existence and uniqueness of finite field with p^n elements is proven for any prime number p and natural number n.
We denote finite field with p^n elements as $\mathbb{F}_{p^n}$. When n=1, $\mathbb{Z}_p$ is isomorphic to $\mathbb{F}_p$. By uniqueness of $\mathbb{F}_p$, we may treat $\mathbb{F}_p$ as $\mathbb{Z}_p$, which we may be more familiar with.
Given $\mathbb{F}_p$, we may extend the field to $\mathbb{F}_p^2$ via field extension. Consider a field extension with $\sqrt{-1} = i$ and we have

$$\text{Span}_{\mathbb{F}_p}\{1, i\} = \{a + bi \mid 0 \leq a, b \leq p - 1\}$$

is a field with p^2 elements and we may denote it as $\mathbb{F}_{p^2}$. Notice that for all non-zero element a in $\mathbb{F}_p$ and for all non-zero element b in $\mathbb{F}_{p^2}$, we have

$$a^{p-1} = 1$$
$$b^{p^2-1} = 1$$

The orders of multiplicative group in the fields of $\mathbb{F}_p$ and $\mathbb{F}_{p^2}$ are $p-1$ and p^2-1 respectively.
We may further extend the finite field to $\mathbb{F}_{p^3}$ and all the way up to $\mathbb{F}_{p^n}$ where n is an natural number. The key idea in extending finite field $\mathbb{F}_p$ to $\mathbb{F}_{p^n}$ is to find some element ω in which the relationship between $\mathbb{F}_p$ and ω is an equation of power n.
If you have learnt about abstract algebra, you may notice that this is constructing fields with irreducible polynomial. If you fell this is too abstract, we again focus on the case of $\mathbb{F}_{p^2}$.
Let $f = t^2 + 1$ and we consider principle ideal domain $\mathbb{F}_p[t]$. We define

$$\mathbb{F}_p[t] = \{f = a_0 + a_1 t + \cdots + a_n t^n \mid n \in \mathbb{N}_0, a_0, a_1, \ldots, a_n \in \mathbb{F}_p\}$$

Notice that f has no root in $\mathbb{F}_p$ (the roots of f is $\pm i$) and that is why we call it irreducible function in $\mathbb{F}_p$. You may skip the proof and take it as granted that $\mathbb{F}_p[t]/\langle f\rangle$ is a field with cardinality p^2. This is how we construction field $\mathbb{F}_{p^2}$. Analogously, we may construction $\mathbb{F}_{p^n}$ for any $n \in \mathbb{N}$ by looking for proper polynomial.

2.2 Elliptic Curve Pairings

2.2.1 Elliptic Curve Pairings

A pairing is a function

$$e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$$

where $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ have prime order p and

- $\mathbb{G}_1$ is typically an additive subgroup of $E(\mathbb{F}_p)$.
- $\mathbb{G}_2$ is typically an additive subgroup of $E(\mathbb{F}_{p^k})$.
- $\mathbb{G}_1$ is typically an multiplicative subgroup of $E(\mathbb{F}^*_{p^k})$.

Note that we call k as embedding degree, which is calculated as the smallest positive integer such that p divides $p^k - 1$.
Two properties of pairing are for cryptography, which are bilinearity and non-degeneracy.
Bilinearity says that for any $u \in \mathbb{G}_1$ and $v, w \in \mathbb{G}_2$, we have

$$e(u, v \cdot w) = e(u, v) \cdot e(u, w)$$

With this property, we may further deduce that for any $u \in \mathbb{G}_1$, $v \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_r$, we have

$$\begin{aligned} e(u^a, v) &= e(u, v)^a \\ e(u, v^b) &= e(u, v)^b \\ e(u^a, v^b) &= e(u, v)^{ab} \end{aligned}$$

Non-degeneracy requires that

$$e(g_1, g_2) \neq \mathbf{1}_{G_T}$$

where g_1 and g_2 are generator of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively.

2.2.2 BLS12-381

We use the popular pairing-friendly BLS12 − 381 curve as an practical example. Note that BLS12 − 381 is also the curve ethereum planed to use for Proto-Danksharding,
For BLS12-381, three groups $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ are defined as follow:

- $\mathbb{G}_1 \subset E(\mathbb{F}_q)$, where $E : y^2 = x^3 + 4$
- $\mathbb{G}_2 \subset E'(\mathbb{F}_{q^2})$, where $E' : y^2 = x^3 + 4(1 + i)$ is called a twisted curve
- $\mathbb{G}_T$ are all the pth unity in $\mathbb{F}_{q^k}$, where $k = 12$ and is the embedding degree

Notice that $\mathbb{G}_2$ here is not a subgroup of $E(\mathbb{F}_{q^{12}})$. This is because doing arithmetic in $\mathbb{F}_{q^{12}}$ is horribly complicated and inefficient. Hence we use some coordinate transformation to transform $\mathbb{F}_{q^{12}}$ into some lower degree field which still has an order p group. We call this process as twist.

2.3 Polynomial

2.3.1 Lagrange Interpolation

We use Lagrange Interpolation to construct a polynomial which passes through n pairs $(x_i, y_i)_{i\in[1,n]}$ we want.
Luckily by Lagrange interpolation polynomial, we may do so. For any $i \in [1, n]$, consider Kronecker delta $\ell_i(X)$ in which

$$\ell_i(X) = \prod_{j\in[1,n],j\neq i} \frac{X-x_j}{x_i-x_j}$$

You may easily check that $\ell_i(X) = 1$ when $X = x_i$ and $\ell_i(X) = 0$ otherwise. With help pf polynomial expression of Kronecker delta, we have Lagrange interpolation polynomial

$$\mathcal{L}(X) = \sum_{i\in[1,n]} y_i \cdot \ell_i(X)$$

Again, you may easily check that n pairs $(x_i, y_i)_{i\in[1,n]}$ lies in the polynomial $\mathcal{L}(X)$.

2.4 Cryptography Assumptions

The security of many schemes in cryptography are unfortunately rely on some assumption. Those assumptions are some problems which we believe it hard however we cannot prove the hardness rigorously. I will introduce the assumptions which we use to prove the security of the KZG commitment.
Discret Logarithm (DL) Assumption. Given a generator g of $\mathbb{G}^*$, where $\mathbb{G}^* = \mathbb{G}$ or $\mathbb{G}_T$, and a is randomly chosen from $\mathbb{Z}_p^*$ for every probabilistic polynomial time (*p.p.t.*) adversary $\mathcal{A}$, probability of calculate a given g and g^a is negligible, i.e.

$$\Pr[\mathcal{A}(g, g^a) = a] = \epsilon(\kappa)$$

where $\epsilon(\cdot)$ is a negligible function and κ is the security parameter.
t-strong Diffie-Hellman (t-SDH) Assumption. Let α randomly chosen from $\mathbb{Z}_p^*$. Given as input a $(t+1)$-tuple $\langle g, g^\alpha, g^{\alpha^2}, \ldots, g^{\alpha^t}\rangle \in \mathbb{G}^{t+1}$, for every p.p.t. adversary $\mathcal{A}$, the probability

$$\Pr[\mathcal{A}(g, g^\alpha, g^{\alpha^2}, \ldots, g^{\alpha^t}) = \langle c, g^{\frac{1}{\alpha+c}}\rangle] = \epsilon(\kappa)$$

for any value of $c \in \mathbb{Z}_p\backslash\{-\alpha\}$.
t-Bilinear strong Diffie-Hellman (t-BSDH) Assumption. Let α randomly chosen from $\mathbb{Z}_p^*$. Given as input a $(t+1)$-tuple $\langle g, g^\alpha, g^{\alpha^2}, \ldots, g^{\alpha^t}\rangle \in \mathbb{G}^{t+1}$, for every p.p.t. adversary $\mathcal{A}$, the probability

$$\Pr[\mathcal{A}(g, g^\alpha, g^{\alpha^2}, \ldots, g^{\alpha^t}) = \langle c, e(g, g)^{\frac{1}{\alpha+c}}\rangle] = \epsilon(\kappa)$$

for any value of $c \in \mathbb{Z}_p\backslash\{-\alpha\}$.

3 KZG Polynomial Commitment Scheme

With the preliminaries, we are now ready to meet KZG polynomial Commitment scheme.

3.1 Trusted Setup

To commit a polynomial with degree $\leq n$, we pick a generator g of some pairing-friendly elliptic curve $\mathbb{G}$ and choose a $\tau \in \mathbb{F}_p$ randomly. Compute $(g, g^\tau, g^{\tau^2}, \ldots, g^{\tau^n})$ and publish it. Notice that we call τ the trapdoor and it should be deleted to guarantee the security of the commitment.

3.2 Commitments

Given a polynomial $\phi(X) = \sum_{i\in[1,l]} \ell_i(X) \cdot X^i$ where $l \leq n$, commitment c to $\phi(X)$ is computed as

$$c = \prod_{i\in[0,l]} (g^{\tau^i})^{\ell_i} = g^{\phi(\tau)}$$

3.3 Evaluation Proofs

To generate a proof for evaluation $\phi(a) = y$, a quotient polynomial $q(X)$ is computed as

$$q(X) = \frac{\phi(X) - y}{X - a}$$

Notice that $q(x)$ is a polynomial **if and only if** $\phi(a) = y$ and we have evaluation proof π in which

$$\pi = g^{q(\tau)}$$

The computation is done in the same manner as c.

3.4 Verification

Verifier is now given the commitment $c = g^{\phi(\tau)}$, the evaluation $y = \phi(a)$ and the proof $\pi = g^{q(\tau)}$. The verification is to check whether

$$e(\frac{c}{g^y}, g) = e(\pi, \frac{g^\tau}{g^a})$$

By bilinearity and non-degeneracy of elliptic curve pairing, we have

$$\begin{aligned} & e(\frac{c}{g^y}, g) = e(\pi, \frac{g^\tau}{g^a}) \\ \Leftrightarrow \quad & e(g^{\phi(\tau)-y}, g) = e(g^{q(\tau)}, g^{\tau-a}) \\ \Leftrightarrow \quad & e(g,g)^{\phi(\tau)-y} = e(g,g)^{q(\tau)(\tau-a)} \\ \Leftrightarrow \quad & \phi(\tau) - y = q(\tau)(\tau - a) \end{aligned}$$

This effectively checks that whether $q(X) = \frac{\phi(X)-y}{X-a}$ holds for $X = \tau$.

3.5 Batch Proofs

We can also prove multiple evaluations $(\phi(e_i) = y_i)_{i \in I}$ for arbitrary points e_i.
We compute

$$q_I(X) = \frac{\phi(X) - R_I(X)}{A_I(X)}$$

where

$$\begin{aligned} A_I(X) &= \prod_{i \in I}(X - e_i) \\ R_I(e_i) &= y_i, \forall i \in I \end{aligned}$$

Note that we may construct such $R_I(e_i)$ via Lagrange interpolation, which is

$$R_I(X) = \sum_{i \in I} y_i \prod_{j \in I, j \neq i} \frac{X - e_j}{e_i - e_j}$$

We compute the batch proof $\pi_I = g^{q_I(\tau)}$.

3.6 Verification of a Batch Proof

Verifier is now given the commitment $c = g^{\phi(\tau)}$, the evaluation $(\phi(e_i) = y_i)_{i \in I}$ and the proof $\pi_I = g^{q_I(\tau)}$.
Verifier does the following:

- Verifier interpolates the accumulator polynomial $A_I(X) = \prod_{i \in I}(X - e_I)$ and computes

$$a = g^{A_I(\tau)}$$

- Verifier interpolates $R_I(X)$ such that $R_I(e_i) = y_i$ as we previously discuss in Section 3.5 and computes

$$r = g^{R_I(\tau)}$$

- Check whether $q_I(X) = \frac{\phi(X)-R_I(X)}{A_I(X)}$ using two pairs $e(\frac{c}{r}, g) = e(\pi_I, a)$. Note that

$$\begin{aligned} & e(\frac{c}{r}, g) = e(\pi_I, a) \\ \Leftrightarrow \quad & e(g^{\phi(\tau)-R_I(\tau)}, g) = e(g^{q_I(\tau)}, g^{A_I(\tau)}) \\ \Leftrightarrow \quad & e(g, g)^{\phi(\tau)-R_I(\tau)} = e(g, g)^{q_I(\tau)A_I(\tau)} \\ \Leftrightarrow \quad & \phi(\tau) - R_I(\tau) = q_I(\tau)A_I(\tau) \end{aligned}$$

This effectively checks that whether $q_I(X) = \frac{\phi(X)-R_I(X)}{A_I(X)}$ holds for $X = \tau$.

3.7 Security Proofs for PolyCommit Scheme

This part will focus on security proofs of single evaluation. Security proofs of multiple polynomials are done in the similar manner but guaranteed by t-BSDH assumption.

3.7.1 Polynomial Binding

Since verifiers use commitment to verify, polycommit scheme have to ensure that it is computationally hard for adversary to find a different polynomial $\phi'(x)$ which will generate same commitment c as $\phi(x)$. Suppose there exists an adversary that breaks polynomial binding for a commitment c by outputting two polynomials $\phi(x)$ and $\phi'(x)$ such that

$$c = g^{\phi(x)} = g^{\phi'(x)}$$

We define polynomial $\phi''(x) := \phi(x) - \phi'(x) \in \mathbb{Z}_p[x]$ and we compute corresponding commitment

$$c' = g^{\phi''(\tau)} = \frac{g^{\phi(\tau)}}{g^{\phi'(\tau)}} = 1$$

therefore $\phi''(\alpha) = 0$, τ is a root of polynomial $\phi''(x)$. Note that we may efficiently find τ via factoring $\phi''(x)$ and finding such $\phi'(x)$ helps us to solve t-SDH problem, which is our security assumption. We hence say such attack is computationally hard as solving t-SDH problem is computationally hard.

3.7.2 Evaluation Binding

Polycommit scheme has to ensure that no adversary can produce y' and π' such that verification is valid. We will construct an algorithm to show that $\mathcal{A}'$ accomplishes such attack helps $\mathcal{A}$ solves t-SDH problem. Suppose there exists an adversary $\mathcal{A}'$ outputs (a, y, π) and (a, y', π') such that verification is valid. By doing some arithmetic on verification formula, we have

$$e(c, g) = e(\pi, g^{(\tau-a)})e(g, g)^y = e(\pi', g^{(\tau-a)})e(g, g)^{y'}$$

Note that $\pi = g^{q(\tau)}$ and $\pi' = g^{q(\tau)'}$, where $q(X)' = \frac{\phi(X)-y'}{X-a}$. Therefore, it is equivalent saying

$$\begin{aligned} q(\tau)(\tau - a) + y &= q(\tau)'(\tau - a) + y' \\ \frac{q(\tau) - q(\tau)'}{y' - y} &= \frac{1}{\tau - a} \\ (\frac{\pi}{\pi'})^{\frac{1}{y'-y}} &= g^{\frac{1}{\tau-a}} \end{aligned}$$

Adversary $\mathcal{A}'$ sends $(\frac{\pi}{\pi'})^{\frac{1}{y'-y}}$ to $\mathcal{A}$ and $\mathcal{A}$ then may returns $\langle -a, (\frac{\pi}{\pi'})^{\frac{1}{y'-y}} (= g^{\frac{1}{\tau-a}}) \rangle$ as a solution for t-SDH problem.

4 References

1. notes on KZG polynomial commitment by Dankfrad Feist
2. notes on KZG polynomial commitment by Alin Tomescu
3. BLS12-381 For The Rest Of Us by Ben Edginton
4. Exploring Elliptic Curve Pairings by Vitalik Buterin
5. KZG in Practice by Andy Arditi
6. Polynomial Commitment by Aniket Kate; Gregory M. Zaverucha; Ian Goldberg