Notes on $\mathcal{PlonK}$: Permutations over Lagrange-bases for Oecumenical Noninteractive Arguments of Knowledge

Alexander Lindenbaum

1 Introduction

These are my notes reading through the celebrated 2019 $\mathcal{PlonK}$ paper, due to [GWC19]. I have written these notes in the order in which I understand the paper, and not the order in which facts are displayed in the paper itself. Keep in mind that these notes are not reviewed by anyone else, and mostly serve as a reference for myself. That being said, any notes or corrections are welcome! Feel free to email me at lindenbaum.alexander@gmail.com with suggestions.

Thank you to Trapdoor-Tech on Medium for providing a secondary reference for reading the $\mathcal{PlonK}$ paper. You can find the reference here.

2 plonk Introduction

This is section 1 of [GWC19].

$\mathcal{PlonK}$ is a zk-SNARK which, similarly to KZG in [KZG10], uses a *structured reference string (SRS)*. One downside to using an SRS is that the SRS must be constructed in a trusted setup, away from any provers. Otherwise, the construction could leak information that an adversarial prover could use to convince verifiers of false statements. The main contribution of $\mathcal{PlonK}$ is a zk-SNARK protocol with a "universal and updatable" SRS. This means that the same SRS can be used for statements about all circuits of a bounded size, and any party can update the SRS in such a manner that the soundness of only one party from all updaters is required for soundness.

$\mathcal{PlonK}$ has a faster prover runtime compared to Sonic, a fully succinct zk-SNARK for circuit sat, due to Maller et al. [MBKM19]. The rest of the notes will explain $\mathcal{PlonK}$ and its improvements.

3 Polynomial protocols for identifying permutations

This is section 5 of [GWC19].

3.1 Checking permutations of univariate polynomials

Let n, d be two integer parameters, with $n \leq d$. Think of n as the degree of the honest prover's polynomials, and d as a bound the verifier enforces on adversarial provers. Also assume we have a prime field $\mathbb{F}$ and $H \subset \mathbb{F}$ is a multiplicative subgroup of order n with generator $\mathbf{g}$.

For $i = 1, \ldots, n$, let $L_i(X) \in \mathbb{F}_{<n}[X]$ such that $L_i(\mathbf{g}^i) = 1$ and $L_i(a) = 0$ for $a \in H$, $a \neq \mathbf{g}^i$. $\{L_i\}_{i \in [n]}$ is called the Lagrange basis for H. One useful fact about the Lagrange basis is that it can "reduce point checks to range checks." What does this mean?

Fact 1. *Fix $i \in [n]$, and $Z, Z^* \in \mathbb{F}[X]$. Then $L_i(a)(Z(a) - Z^*(a)) = 0$ for each $a \in H$ if and only if $Z(\mathbf{g}^i) = Z^*(\mathbf{g}^i)$.*

Proof. 1. Suppose $Z(\mathbf{g}^i) = Z^*(\mathbf{g}^i)$. If $a = \mathbf{g}^i$ then $L_i(a)(Z(a) - Z^*(a))$ becomes $1 \cdot 0 = 0$. If $a \neq \mathbf{g}^i$ then $L_i(a) = 0$.

2. Suppose for all $a \in H$ that $L_i(a)(Z(a) - Z^*(a)) = 0$. Then

$$L_i(\mathbf{g}^i)(Z(\mathbf{g}^i) - Z^*(\mathbf{g}^i)) = Z(\mathbf{g}^i) - Z^*(\mathbf{g}^i) = 0.$$

□

So to check if two polynomials Z, Z^* are equal at $\mathbf{g}^i$, it is enough to check that the range of $L_i(X)(Z(X) - Z^*(X))$ is 0 on H.

For $f, g \in \mathbb{F}_{<d}[X]$ and a permutation $\sigma : [n] \to [n]$, we say $g = \sigma(f)$ if for all $i \in [n]$, $g(\mathbf{g}^i) = f(\mathbf{g}^{\sigma(i)})$. That is, f simply permutes the list $\langle g(\mathbf{g}^1), g(\mathbf{g}^2), \ldots, g(\mathbf{g}^n) \rangle$.

Here is a "ranged polynomial" protocol for a polytime prover P_{poly} to prove that $g = \sigma(f)$.

Preprocessed polynomials Computed before the protocol is executed:

- $S_{\text{ID}} \in \mathbb{F}_{<n}[X]$ defined by $S_{\text{ID}}(\mathbf{g}^i) = i \ \forall i \in [n]$.
- $S_\sigma \in \mathbb{F}< n[X]$ defined by $S_\sigma(\mathbf{g}^i) = \sigma(i) \ \forall i \in [n]$.

Inputs $f, g \in \mathbb{F}_{<n}[X]$.

Protocol

1. V_{poly} chooses two random field elements β, γ and sends them to P_{poly}.
2. Let

$$f' := f + \beta \cdot S_{\text{ID}} + \gamma,$$
$$g' := g + \beta \cdot S_\sigma + \gamma.$$

So for all $i \in [n]$,

$$f'(\mathbf{g}^i) = f(\mathbf{g}^i) + \beta \cdot i + \gamma,$$
$$g'(\mathbf{g}^i) = g(\mathbf{g}^i) + \beta \cdot \sigma(i) + \gamma.$$

3. P_{poly} computes $Z \in \mathbb{F}_{<n}[X]$ such that $Z(\mathbf{g}) = 1$ and for $i \in \{2, \ldots, n\}$,

$$Z(\mathbf{g}^i) = \prod_{1 \le j < i} f'(\mathbf{g}^j)/g'(\mathbf{g}^j)$$

(if one of $Z(\mathbf{g}^i)$ is undefined, the protocol is aborted. This happens with negligible probability).

4. P_{poly} sends Z to V_{poly}.

5. V_{poly} checks if for all $a \in H$ that

   (a) $L_1(a)(Z(a) - 1) = 0$,

   (b) $Z(a)f'(a) = g'(a)Z(a \cdot \mathbf{g})$

   and accepts if all checks hold.

Correctness If $g = \sigma(f)$ for some permutation σ, then we just have to show that both checks (a) and (b) pass with P_{poly} is honest.

(a) holds if and only if $Z(\mathbf{g}) = 1$, which is the case under an honest prover.

As for (b), observe that as we defined Z:

$$Z(\mathbf{g}^{i+1}) = Z(\mathbf{g}^i) \cdot \frac{f'(\mathbf{g}^i)}{g'(\mathbf{g}^i)}.$$

Rearrange the terms and we are done.

Soundness This relies on the Shwartz-Zippel lemma, exemplified in the following lemma (Claim A.1 in the paper:

Lemma 2. *Fix a permutation $\sigma : [n] \to [n]$, and say $a_1, \ldots, a_n, b_1, \ldots, b_n \in \mathbb{F}$. If for uniformly random chosen $\beta, \gamma \in \mathbb{F}$ we have*

$$\prod_{i \in [n]} (a_i + \beta \cdot i + \gamma) = \prod_{i \in [n]} (b_i + \beta \cdot \sigma(i) + \gamma)$$

with probability $> n/|\mathbb{F}|$, then for all $i \in [n]$ $b_i = a_{\sigma(i)}$ (i.e. the polynomials on both sides are equal).

Now suppose that $g \neq \sigma(f)$. By Lemma 2, except with negligible probability over choice of β, γ, we have

$$a := \prod_{i \in [n]} f'(\mathbf{g}^i) \neq b := \prod_{i \in [n]} g'(\mathbf{g}^i).$$

Assume indeed that $a \neq b$ and that $g'(\mathbf{g}^i) \neq 0$ ever. We show that if both checks pass then we have a contradiction.

We know $Z(\mathbf{g}) = 1$ from (a). We also know from (b) that

$$Z(\mathbf{g}^{i+1}) = \prod_{i \le j < i} \frac{f'(\mathbf{g}^j)}{g'(\mathbf{g}^j)}.$$

Then $Z(\mathbf{g}^{n+1}) = a/b \neq 1$. But $\mathbf{g}^{n+1} = \mathbf{g}$ and so we have a contradiction.

3.2 "Extended" permutations and copy-satisfy

There is a way to extend the meaning of $g = \sigma(f)$ for many polynomials $f_1, \ldots$ and $g_1, \ldots$, then check for this property. I will define this, but not go through the protocol for "extended" permutations. All I will say is that it is very similar to the case of single polynomials, as above.

Say we have $f_1, \ldots, f_k, g_1, \ldots, g_k \in \mathbb{F}_{<d}[X]$. For each position in $[kn]$ we are going to define a value corresponding to f and one corresponding to g. Specifically, for $j \in [k]$ (block) and $i \in [n]$ (position within block j), we define

$$f_{((j-1)\cdot n+1)} = f_j(\mathbf{g}^i),$$
$$g_{((j-1)\cdot n+1)} = g_j(\mathbf{g}^i).$$

This is just saying that we are evaluating all polynomials over all $\mathbf{g}^i$. Let $\sigma : [kn] \to [kn]$ be a permutation over the kn positions. We write that

$$(g_1, \ldots, g_k) = \sigma(f_1, \ldots, f_k)$$

if

$$\forall \ell \in [kn],\ g_{(\ell)} = f_{(\sigma(\ell))}.$$

As said before, there is a similar interactive protocol for testing for this property.

Crypto primitive of importance Let $\mathcal{T} = \{T_1, \ldots, T_s\}$ be a partition of $[kn]$ into disjoint blocks. For $f_1, \ldots, f_k \in \mathbb{F}_{<n}[X]$, we say that $(f_1, \ldots, f_k)$ *copy-satisfy* $\mathcal{T}$ if $f_{(\ell)} = f_{(\ell')}$ whenever ℓ, ℓ' are in the same block of $\mathcal{T}$.

We can use our test for extended permutations to test if $(f_1, \ldots, f_k)$ copy-satisfy $\mathcal{T}$ in the following way: define a permutation π on $[kn]$ such that for each block T_i, σ contains a cycle going over all elements of T_i. Then $(f_1, \ldots, f_k)$ copy-satisfy $\mathcal{T}$ if and only if $(f_1, \ldots, f_k) = \sigma(f_1, \ldots, f_k)$.

We will come back to this tool when we put $\mathcal{P}\mathfrak{lon}\mathcal{K}$ all together.

4 Constraint systems

This is section 6 of [GWC19].

Constraint systems are as they sound: a means to model the contraints of some system. In this case, constraint systems capture arithmetic circuits with fan-in two and unlimited fan-out, n gates and m wires. But they are more general. All zk-SNARKS need a computational problem which they solve, and a way of encoding statements we would like to prove. Constraint systems are one such way.

Since I do not know how to use tikz... I will suggest a reference to see how constraint systems simulate arithmetic circuits. Vitalik Buterin has a great diagram and explanation, link here. Constraint systems are also a satisfaction problem.

Fix positive integers m and n. A constraint system $\mathscr{C} = (\mathcal{V}, \mathcal{Q})$ is defined as folows:

- $\mathcal{V} = (\mathbf{a}, \mathbf{b}, \mathbf{c})$, where $\mathbf{a}, \mathbf{b}, \mathbf{c} \in [m]^n$. This specifies, for each gate $i \in [n]$, what are the "left," "right," and "output" wires.
- $\mathcal{Q} = (\mathbf{q_L}, \mathbf{q_R}, \mathbf{q_O}, \mathbf{q_M}, \mathbf{q_C}) \in (\mathbb{F}^n)^5$. These select whether gates are addition or multiplication gates.

$\mathbf{x} \in \mathbb{F}^{\mathbf{m}}$ *satisfies* $\mathscr{C}$ if for each $i \in [n]$,

$$(\mathbf{q_L})_i \cdot \mathbf{x_{a_i}} + (\mathbf{q_R})_i \cdot \mathbf{x_{b_i}} + (\mathbf{q_O})_i \cdot \mathbf{x_{c_i}} + (\mathbf{q_M})_i \cdot (\mathbf{x_{a_i}} \mathbf{x_{b_i}}) + (\mathbf{q_C})_i = 0.$$

To define a relation on $\mathscr{C}$, we let $\ell \leq m$ be some positive integer, and consider $\mathcal{I} = \{1, \ldots, \ell\}$ the set of "public inputs." Then the relation $\mathcal{R}_{\mathscr{C}}$ is the set of pairs (x, ω) with $x \in \mathbb{F}^\ell$ and $\omega \in \mathbb{F}^{m-\ell}$ such that (x, ω) satisfies $\mathscr{C}$.

Arithmetic Circuits The paper has a reduction from arithmetic circuits to constraint systems in section 6. It is worth checking that it works. We can also enforce that our witness must take 0/1 values, by altering constraints.

5 Idealized low-degree protocols

This is section 4 of [GWC19]. I have already taken notes on the KZG protocol.

We specify a generalization of a polynomial commitment scheme like KZG. This is a protocol between prover, verifier, and a trusted party $\mathcal{I}$, which acheives the following. The prover sends low-degree polynomials to $\mathcal{I}$. Then, the verifier may ask $\mathcal{I}$ whether certain identities hold between those polynomials, and additional predefined polynomials known to the verifier.

Let d, D, t, ℓ be positive integers. Consider the following protocol, called a (d, D, t, ℓ)-polynomial protocol.

1. The protocol already has baked in preprocessed polynomials $g_1, \ldots, g_\ell \in \mathbb{F}_{<d}[X]$.
2. P_{poly} sends messages to $\mathcal{I}$ of the form f for $f \in \mathbb{F}_{<d}[X]$. If the messages are not of this form, we abort.

3. We are in the public coin model, so V_{poly}'s messages to P_{poly} are random coins.

4. At the end of the protocol, suppose $f_1, \ldots, f_t$ are the polynomials sent from P_{poly} to $\mathcal{I}$. V_{poly} asks $\mathcal{I}$ if polynomial identites hold between $\{f_1, \ldots, f_t, g_1, \ldots, g_\ell\}$, where each identity is of the form
$$F(X) := G(X, h_1(v_1(X)), \ldots, h_M(v_M(X))) \equiv 0,$$
for some $h_i \in \{f_1, \ldots, f_t, g_1, \ldots, g_\ell\}$, $G \in \mathbb{F}[X, X_1, \ldots, X_M]$, $v_1, \ldots, v_M \in \mathbb{F}_{<d}[X]$ such that $F \in \mathbb{F}_{<D}[X]$ for every choice of $f_1, \ldots, f_t$ made by P_{poly} when following the protocol honestly.

5. After receiving answers from $\mathcal{I}$, V_{poly} accepts if all identities hold, otherwise rejects.

Definition 3. Given a relation $\mathcal{R}$, a polynomial protocol for $\mathcal{R}$ is a polynomial protocol with the following additional properties:

1. P_{poly} and V_{poly} are both given x. If there is an ω where $(x, \omega) \in \mathcal{R}$, then we assume P_{poly} has possession of ω.

2. Completeness: If P_{poly} follows the protocol correctly using a witness ω for x, V_{poly} accepts w.p. 1.

3. Knowledge Soundness: There exists an efficient algorithm E, that given access to the messages of P_{poly} and $\mathcal{I}$ outputs ω such that, for any strategy of P_{poly}, the probability that both:

   (a) V_{poly} accepts, and
   (b) $(x, \omega) \notin \mathcal{R}$

   is negligible.

5.1 Polynomial protocols on ranges

There is a distinction we can make where V_{poly} only needs to check if the polynomial equations hold on a certain range of input values. We call this an S-ranged (d, D, t, ℓ)-polynomial protocol. In fact, we already wrote an H-ranged protocol for testing if $(g_1, \ldots, g_k) = \sigma(f_1, \ldots, f_k)$.

Lemma 4. *Let $\mathscr{P}$ be an S-ranged (d, D, t, ℓ)-polynomial protocol for $\mathcal{R}$. Then we can construct a $(\max\{d, |S|, D - |S|\}, D, t + 1, \ell + 1)$-polynomial protocol $\mathscr{P}^*$ for $\mathcal{R}$.*

This says that we can convert a ranged protocol to a polynomial protocol, and only incur one additional prover polynomial! We are also using a particular SRS string here.

The culmination of section 4 the last lemma, which states that if we have a public coin polynomial protocol for a relation $\mathcal{R}$, then we can construct a protocol for $\mathcal{R}$ with knowledge soundness in the Algebraic Group Model, under a $2d$-DLOG assumption. I won't write the proof, but I will expand on what this means.

- The SRS here is a string of monomials over the field $\mathbb{F}$ with coefficients in two groups of the same order, evaluated at a randomly chosen point $x \in \mathbb{F}$. The Q-DLOG assumption for groups $(\mathbb{G}_1, \mathbb{G}_2)$ states that given an SRS string as above, where $x \in \mathbb{F}$ is random, the probability of an efficient $\mathcal{A}$ outputting x is negligible. So no efficient prover can recover the secret parameter used to create the SRS, in order to fool verifiers.

- The Algebraic Group Model (AGM) was introduced in [FKL18]. It models adversaries which can output field elements to participate in the protocol, but they must also output coefficients of the SRS used to represent that field element. This is harder to be secure in than the Generic Group Model (GGM), which prior SNARK constructions base their security in.

6 The main protocol

This is section 7 of [GWC19].

Let $\mathscr{C} = (\mathcal{V}, \mathcal{Q})$ be a constraint system. Here is a protocol for the relation $\mathcal{R}_{\mathscr{C}}$. First, we define a partition of $\mathscr{C}$.

Suppose $\mathcal{V} = (\mathbf{a}, \mathbf{b}, \mathbf{c})$. Think of $\mathcal{V}$ as a vector in $[m]^{3n}$. For $i \in [m]$, let $T_i \subset [3n]$ be the set of indices $j \in [3n]$ such that $V_j = i$. Now define

$$\mathcal{T}_{\mathscr{C}} := \{T_i\}_{i \in [m]}.$$

Here is an H-ranged polynomial protocol for $\mathcal{R}_{\mathscr{C}}$.

Protocol

1. Let $x \in \mathbb{F}^m$ be P_{poly}'s input. P_{poly} computes three polynomials $f_L, f_R, f_O \in \mathbb{F}_{<n}[X]$, where for $i \in [n]$
$$f_L(\mathbf{g}^i) = \mathbf{x_{a_i}}, f_R(\mathbf{g}^i) = \mathbf{x_{b_i}}, f_O(\mathbf{g}^i) = \mathbf{x_{c_i}}.$$
P_{poly} sends f_L, f_R, F_O to $\mathcal{I}$.

2. P_{poly} and V_{poly} run the extended permutation check protocol using the permutation between (f_L, f_R, f_O) and itself. As we saw, thick checks if (f_L, f_R, f_O) copy-satisfies $\mathcal{T}_{\mathscr{C}}$.

3. V_{poly} computes the "Public input polynomial"
$$\text{PI}(X) := \sum_{i \in [\ell]} -x_i \cdot L_i(x).$$

4. V_{poly} now checks the identity
$$\mathbf{q_L} \cdot f_L + \mathbf{q_R} \cdot f_R + \mathbf{q_O} \cdot L_O + \mathbf{q_M} \cdot f_L \cdot f_R + (\mathbf{q_C} + \text{PI}) = 0,$$
on H.

Theorem 5. *The protocol is an H-ranged polynomial protocol for the relation $\mathcal{R}_{\mathscr{C}}$.*

Proof. If (f_L, f_R, f_O) copy satisfies $\sigma(\mathscr{C})$ and the identity checked by V_{poly} holds, then $(x, \omega) \in \mathcal{R}_{\mathscr{C}}$.

Knowledge soundness follows from describing an extractor E. E uses the values of f_L, f_R, f_O to constuct an assignment naturally: if $\mathbf{a}_i = j$, let $\mathbf{x}_j = f_L(\mathbf{g}^i)$. E then outputs the last $m - \ell$ elements for a witness ω.

Look at the event where $(x, \omega) \notin \mathcal{R}$ but V_{poly} still accepts. This event has two subevents: one where (f_L, f_R, f_O) doesn't copy-satisfy $\sigma(\mathscr{C})$ and the one where it does. The first event has negligible probability. The second case where the triple does copy-satisfy $\sigma(\mathscr{C})$ and last identity holds. In fact, it must be the case that $(x, \omega) \in \mathcal{R}$. So the whole event has negligible probability. □

The paper already showed that a S-ranged polynomial protocol for the relation yields a protocol for the relation with Knowledge Soundness in the AGM. Applying that for $\mathcal{R}_{\mathscr{P}}$, we get $\mathcal{P}\mathfrak{lonK}$. Moreover, they compute that the prover requires $11n + 1$ group operations, and the total prover communication consists of 7 group elements and 7 field elements.

7 Some Comments in the Paper

- We did not talk about zero knowledge yet. They claim that all that has to be done to add zero knowledge is random multiples of Z_H to the prover polynomials, and requiring the verifier to send challenges in $\mathbb{F} \setminus H$.
- H is the multiplicative subgroup containing the n'th roots of unity in $\mathbb{F}$.
- The last thing that needs to be done is to tranform $\mathcal{P}\mathfrak{lonK}$ into a non-interactive proof, by applying Fiat-Shamir.

References

[FKL18] Georg Fuchsbauer, Eike Kiltz, and Julian Loss. The algebraic group model and its applications. In *Advances in Cryptology–CRYPTO 2018: 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19–23, 2018, Proceedings, Part II 38*, pages 33–62. Springer, 2018.

[GWC19] Ariel Gabizon, Zachary J Williamson, and Oana Ciobotaru. Plonk: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. *Cryptology ePrint Archive*, 2019.

[KZG10] Aniket Kate, Gregory M Zaverucha, and Ian Goldberg. Constant-size commitments to polynomials and their applications. In *Advances in Cryptology-ASIACRYPT 2010: 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings 16*, pages 177–194. Springer, 2010.

[MBKM19] Mary Maller, Sean Bowe, Markulf Kohlweiss, and Sarah Meiklejohn. Sonic: Zero-knowledge snarks from linear-size universal and updatable structured reference strings. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, page 2111–2128, New York, NY, USA, 2019. Association for Computing Machinery.