Special thanks to Zac and Ariel for answering numerous questions, and for pointing out that neighbouring PLONK gates can share wires through shifts.

TLDR

: We suggest a simplification to PLONK called SLONK. We replace the permutation argument (the “P” in PLONK) in favour of a shift argument (the “S” in SLONK). We get a universal SNARK with the smallest known proof size and verification time.

Despite a less favourable prover cost model which includes wire length, the prover complexity has improved constants relative to PLONK. The improved constants may lead to faster provers, especially for large circuits where FFTs dominate, or for circuits that can be expressed with little wiring.

Construction part 1—SLONK grid

The SLONK arithmetisation places field elements on a 3D grid. The grid has width, depth and height n_w

, n_d

, n_h

for a total of $n_w n_d n_h$

grid values. We label the grid value at coordinates (i, j, k)

by $v_{i, j, k}$

.

We encode the grid as a polynomial using the Lagrange basis. Specifically, the grid polynomial is $\mathbf{g}(x) = \sum_{i = 1}^{n_w}\sum_{j = 1}^{n_d}\sum_{k = 1}^{n_h} v_{i, j, k}L_{i + j n_w + k n_w n_d}(x)$

.

We also define three shifts $\mathbf{g}_w(x) = \mathbf{g}(x\omega)$

, $\mathbf{g}_d(x) = \mathbf{g}(x\omega^{n_w})$

, $\mathbf{g}_h(x) = \mathbf{g}(x\omega^{n_wn_d})$

. Notice that $\mathbf{g}_w = \sum_{i = 1}^{n_w}\sum_{j = 1}^{n_d}\sum_{k = 1}^{n_h} v_{i-1, j, k}L_{i + j n_w + k n_w n_d}$

shifts the width coordinate i

by 1. Likewise $\mathbf{g}_d$

and $\mathbf{g}_h$

shift the j

and k

coordinates by 1.

Construction part 2—SLONK equation

Given grid coordinates (i, j, k)

let $S_{i, j, k}$

to be set of four points consisting of $v_{i, j, k}$

and its three shifts $v_{i-1, j, k}$

, $v_{i, j-1, k}$

, $v_{i, j, k-1}$

. We now define relations between the points in $S_{i, j, k}$

using $\mathbf{g}$

, $\mathbf{g}_w$

, $\mathbf{g}_d$

, $\mathbf{g}_h$

and six selector polynomials:

$$\mathbf{g}\mathbf{q} + \mathbf{g}_w\mathbf{q}_w + \mathbf{g}_d\mathbf{q}_d + \mathbf{g}_h\mathbf{q}_h + \mathbf{g}\mathbf{g}_w \mathbf{q}_m + \mathbf{q}_c = 0$$

There are four linear selectors $\mathbf{q}$

, $\mathbf{q}_w$

, $\mathbf{q}_d$

, $\mathbf{q}_h$

corresponding to the four points in $S_{i, j, k}$

. These can be configured for addition gates as well as wiring (see below). Similar to PLONK there is one multiplication selector $\mathbf{q}_m$

for multiplication gates, and one constant selector $\mathbf{q}_c$

for public inputs.

Discussion part 1—wiring

The SLONK equation allows for any two points of a given $S_{i, j, k}$

to be wired up. For example, setting $(\mathbf{q})_{i, j, k} = 1$

, $(\mathbf{q}_w)_{i, j, k} = -1$

, and zeroing the other selectors corresponds to a wire connecting $v_{i, j, k}$

to $v_{i-1, j, k}$

.

There are 6 types of short wire segments: three “straight” types that follow the grid lines, and three “diagonal” types. These local wire segments can be concatenated for custom wiring.

SLONK’s wire routing is analogous to the routing of physical wires in electronics. For example, printed circuit boards and ASICs have wires etched as a concatenation of straight wire segments following grid lines, with vertical wires connecting layers to form a 3D network of wires.

Notice that the SLONK grid boundaries wrap around, creating wiring shortcut opportunities not present in physical space. Notice also that routing in electronics is usually done with a small number of layers (say, ~10) because each layer of silicon and metal has significant cost. With SLONK the grid does not have to be flat. For example, it could be shaped as a cube to help gate placement and wire routing.

Discussion part 2—performance

proof size

(BN254)

verifier

$\mathbb{G}_1$

exp

prover

$\mathbb{G}_1$

exp

prover

$\mathbb{F}$

operations

SRS

$\mathbb{G}_1$

size

SLONK (small)

$6 \mathbb{G}_1 + 5\mathbb{F}$

544 bytes

16

$8(a + w)$

$\approx23(a + w)\log(a + w)$

$2(a + w)$

SLONK (fast)

$7 \mathbb{G}_1 + 5\mathbb{F}$

608 bytes

17

$7(a + w)$

$\approx23(a + w)\log(a + w)$

$a + w$

PLONK (small)

$7 \mathbb{G}_1 + 7\mathbb{F}$

672 bytes

16

$11a$

$\approx56a\cdot \log(a)$

$3a$

PLONK (fast)

$9 \mathbb{G}_1 + 7\mathbb{F}$

800 bytes

18

$9a$

$\approx56a\cdot \log(a)$

a

The SLONK proof size is ~20% smaller than PLONK. When using BN254 on Ethereum 1.0 SLONKs are 128 bytes smaller than PLONKs (or 192 bytes smaller when optimising for prover time). SLONK slightly improves verification time over PLONK when optimising for prover speed.

The SLONK and PLONK provers are not directly comparable as they have different cost models. Specifically, only the number a

of arithmetic (i.e. addition and multiplication) gates count for PLONK, whereas for SLONK the wire length w

also counts.

We speculate that the SLONK prover could be faster than the PLONK prover for some practical circuits thanks to the improved constants, especially for large circuits where FFT costs dominate. Indeed, SLONK's FFT constant is 2.5x smaller than PLONK's FFT constant. (Asymptotically FFTs dominate prover time versus multiexponentiations by a $\log^2$ factor.)

Appendix 1—proof sizes breakdown

PLONK

- 3 $\mathbb{G}_1$ elements for the commitment to wire polynomials $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$
- 1 $\mathbb{G}_1$ element for the commitment to the permutation polynomial $\mathbf{z}$
- 1 $\mathbb{G}_1$ element for the commitment to the quotient polynomial $\mathbf{t}$
- 2 $\mathbb{G}_1$ elements for the evaluation points z, $z\omega$
- 6 $\mathbb{F}$ elements $\mathbf{a}(z)$, $\mathbf{b}(z)$, $\mathbf{c}(z)$, $\mathbf{s}_{\sigma 1}(z)$, $\mathbf{s}_{\sigma 2}(z)$, $\mathbf{z}(z\omega)$ for the linearisation polynomial $\mathbf{r}$
- 1 $\mathbb{F}$ element $\mathbf{r}(z)$ for the opening of the linearisation polynomial

SLONK

- 1 $\mathbb{G}_1$ element for the commitment to the grid polynomial $\mathbf{g}$
- 1 $\mathbb{G}_1$ element for the commitment to the quotient polynomial $\mathbf{t}$
- 4 $\mathbb{G}_1$ elements for the evaluation points z, $z\omega$, $z\omega^{n_w}$, $z\omega^{n_wn_d}$
- 4 $\mathbb{F}$

elements $\mathbf{g}(z)$

, $\mathbf{g}_w(z)$

, $\mathbf{g}_d(z)$

, $\mathbf{g}_h(z)$

for the linearisation polynomial $\mathbf{r}$

- 1 $\mathbb{F}$

element $\mathbf{r}(z)$

for the opening of the linearisation polynomial

Appendix 2—prover FFTs breakdown

PLONK

- 12 degree 1a

iFFTs for $\mathbf{q}_m$

, $\mathbf{q}_l$

, $\mathbf{q}_r$

, $\mathbf{q}_o$

, $\mathbf{q}_c$

, $\mathbf{a}$

, $\mathbf{b}$

, $\mathbf{c}$

, $\mathbf{s}_{\sigma 1}$

, $\mathbf{s}_{\sigma 2}$

, $\mathbf{s}_{\sigma 3}$

- 5 degree 2a

FFTs for $\mathbf{q}_m$

, $\mathbf{q}_l$

, $\mathbf{q}_r$

, $\mathbf{q}_o$

, $\mathbf{q}_c$

- 1 degree 2a

iFFT for degree-2a

terms of the quotient polynomial $\mathbf{t}$

- 7 degree 4a

FFTs for $\mathbf{a}$

, $\mathbf{b}$

, $\mathbf{c}$

, $\mathbf{s}_{\sigma 1}$

, $\mathbf{s}_{\sigma 2}$

, $\mathbf{s}_{\sigma 3}$

, $\mathbf{z}$

- 1 degree 4a

iFFT for degree-3a

terms of the quotient polynomial $\mathbf{t}$

SLONK

- 7 degree 1(a + w)

iFFTs for $\mathbf{q}$

, $\mathbf{q}_w$

, $\mathbf{q}_d$

, $\mathbf{q}_h$

, $\mathbf{q}_m$

, $\mathbf{q}_c$

, $\mathbf{g}$

- 7 degree 2(a + w)

FFTs for $\mathbf{q}$

, $\mathbf{q}_w$

, $\mathbf{q}_d$

, $\mathbf{q}_h$

, $\mathbf{q}_m$

, $\mathbf{q}_c$

, $\mathbf{g}$

- 1 degree 2(a + w)

iFFT for the quotient polynomial $\mathbf{t}$

Appendix 3—prover $\mathbb{G}_1$

exponentiations breakdown

PLONK (small)

- 3a

$\mathbb{G}_1$

exponentiations for the wire polynomials $\mathbf{a}$

, $\mathbf{b}$

, $\mathbf{c}$

- 1a

$\mathbb{G}_1$

exponentiations for the permutation polynomial $\mathbf{z}$

- 3a

$\mathbb{G}_1$

exponentiations for the quotient polynomial $\mathbf{t}$

- 3a

$\mathbb{G}_1$

exponentiations for the evaluation at z

- 1a

$\mathbb{G}_1$

exponentiations for the evaluation at $z\omega$

PLONK (fast)

- same as above with 1a

instead of 3a

for the evaluation at z

SLONK (small)

- 1(a + w)

$\mathbb{G}_1$

exponentiations for the grid polynomial $\mathbf{g}$

- 2(a + w)

$\mathbb{G}_1$

exponentiations for the quotient polynomial $\mathbf{t}$

- 2(a + w)

$\mathbb{G}_1$

exponentiations for the evaluation at z

- 1(a + w)

$\mathbb{G}_1$

exponentiations for the evaluation at $z\omega$

- 1(a + w)

$\mathbb{G}_1$

exponentiations for the evaluation at $z\omega^{n_w}$

- 1(a + w)

$\mathbb{G}_1$

exponentiations for the evaluation at $z\omega^{n_wn_d}$

SLONK (fast)

- same as above with 1(a + w)

instead of 2(a + w)

for the evaluation at z

Appendix 4—verifier $\mathbb{G}_1$

exponentiations breakdown

PLONK (small)

- 5 for selectors $[\mathbf{q}_m]_1$

, $[\mathbf{q}_l]_1$

, $[\mathbf{q}_r]_1$

, $[\mathbf{q}_o]_1$

, $[\mathbf{q}_c]_1$

- 3 for wire values $[\mathbf{a}]_1$

, $[\mathbf{b}]_1$

, $[\mathbf{c}]_1$

- 3 for permutations $[\mathbf{z}]_1$

, $[\mathbf{s}_{\sigma 1}]_1$

, $[\mathbf{s}_{\sigma 2}]_1$

- 3 for evaluation points $[\mathbf{W}_{z}]_1$

, $[\mathbf{W}_{z\omega}]_1$

(2x)

- 1 for the batch evaluation $[\mathbf{1}]_1$
- 1 for the quotient polynomial $[\mathbf{t}]_1$

PLONK (fast)

- same as above with $[\mathbf{t}]_1$

replaced by $[\mathbf{t}_{lo}]_1$

, $[\mathbf{t}_{mid}]_1$

, $[\mathbf{t}_{hi}]_1$

SLONK (small)

- 6 for selectors $[\mathbf{q}]_1$

, $[\mathbf{q}_w]_1$

, $[\mathbf{q}_d]_1$

, $[\mathbf{q}_h]_1$

, $[\mathbf{q}_m]_1$

, $[\mathbf{q}_c]_1$

- 1 for grid values $[\mathbf{g}]_1$
- 7 for evaluation points $[\mathbf{W}_{z}]_1$

, $[\mathbf{W}_{z\omega}]_1$

(2x), $[\mathbf{W}_{z\omega^{n_w}}]_1$

(2x), $[\mathbf{W}_{z\omega^{n_wn_d}}]_1$

(2x)

- 1 for the batch evaluation $[\mathbf{1}]_1$
- 1 for the quotient polynomial $[\mathbf{t}]_1$

SLONK (fast)

- same as above with $[\mathbf{t}]_1$

replaced by $[\mathbf{t}_{lo}]_1$

, $[\mathbf{t}_{hi}]_1$