Sangria: A Folding Scheme for PLONK

Nicolas Mohnblatt
Geometry
nico@geometry.xyz

March 6, 2023

Abstract

As shown in Nova [KST21], incrementally verifiable computation (IVC) can be realised using a folding scheme and a zkSNARK. In this technical note, we present a folding scheme for a variant of the PLONK arithmetization [GWC19]. We then extend our relaxed PLONK arithmetization to accept custom gates of degree 2 and circuits with higher gate arity. Finally we outline avenues for future work including folding higher degree gates, supporting lookup gates and designing an IOP for the relaxed PLONK arithmetization.

Contents

1 Preliminaries

We first cover some notation, preliminary knowledge of the PLONK arithmetization and folding schemes.

Notation. We denote column vectors using lowercase boldface letter $\mathbf{x}$ and denote $\mathbf{x}_i$ the i-th element of $\mathbf{x}$. The $\circ$ operator is used for element-wise multiplication; the $||$ operator denotes column-wise concatenation. Matrices are denoted using uppercase boldface letters, $\mathbf{M}$, with $\mathbf{M}_{i,j}$ being the element at the i-th row and j-th column of the matrix.

Our scheme makes use of a hiding and binding additively homomorphic commitment scheme for vectors of elements in a finite field $\mathbb{F}$. We denote such a scheme as Com and write $\overline{A} = \mathsf{Com}\left(\mathsf{pp}_C, \mathbf{a}; r\right)$ a commitment to the vector $\mathbf{a}$ using randomness value $r \in \mathbb{F}$ and commitment parameters pp_C.

PLONK arithmetization. PLONK [GWC19] circuits are defined using multiple instantiations of a single multi-purpose gate. Each instantiation of the gate can be specialised into implementing a desired operation using so-called selectors. The i-th gate is defined by the equation:

$$C_{\mathcal{Q},i}(\mathbf{a},\mathbf{b},\mathbf{c}) := (\mathbf{q_L})_i\mathbf{a}_i + (\mathbf{q_R})_i\mathbf{b}_i + (\mathbf{q_O})_i\mathbf{c}_i + (\mathbf{q_M})_i\mathbf{a}_i\mathbf{b}_i + (\mathbf{q_C})_i \tag{1}$$

where $(\mathbf{q_L})_i, (\mathbf{q_R})_i, (\mathbf{q_O})_i, (\mathbf{q_M})_i, (\mathbf{q_C})_i$ are the i-th value of each selector vector and $\mathbf{a}_i, \mathbf{b}_i, \mathbf{c}_i$ are the left input, right input and output of the gate, as we will see shortly. We denote $\mathcal{Q} = \{\mathbf{q_L}, \mathbf{q_R}, \mathbf{q_O}, \mathbf{q_M}, \mathbf{q_C}\}$ the set of selector vectors.

A circuit is fully defined by s gates (via the set $\mathcal{Q}$ of selector vectors) and a set of copy constraints. The latter ensure that gates are correctly connected, e.g. "the left input of the 5th gate must equal the output of the 4th gate".

Finally, a computation is represented as a table with three columns and $n + s + 1$ rows where "the first n rows encode the n public inputs; the next s rows represent the left and right inputs and the output for each gate; and the final row enforces that the final output of the circuit is zero" [CBBZ22]. We often refer to this table as the *trace* and its columns as the vectors $\mathbf{a}, \mathbf{b}, \mathbf{c}$ (each of length $n + s + 1$).

We can define an *instance* $\mathbf{X}$ and *witness* $\mathbf{W}$ as subtables of the trace:

$$\mathbf{X} := \left\{(\mathbf{a}_i, \mathbf{b}_i, \mathbf{c}_i) \in \mathbb{F}^3\right\}_{i\in[0,n]} \tag{2}$$

$$\mathbf{W} := \left\{(\mathbf{a}_i, \mathbf{b}_i, \mathbf{c}_i) \in \mathbb{F}^3\right\}_{i\in[n+1,n+s]} \tag{3}$$

As notation short-hand, for $k \in \{a, b, c\}$ we define the vectors $\mathbf{x_k} := (\mathbf{k}_0, \mathbf{k}_1, \ldots, \mathbf{k}_n)$ and $\mathbf{w_k} := (\mathbf{k}_{n+1}, \mathbf{k}_{n+2}, \ldots, \mathbf{k}_{n+s+1})$.

Folding schemes. Nova [KST21] introduces the notion of a folding scheme. The paper provides the following intuitive definition:

> [...] a folding scheme for a relation $\mathcal{R}$ is a protocol that reduces the task of checking two instances in $\mathcal{R}$ to the task of checking a single instance in $\mathcal{R}$.

The full definition is given in Definition 6 of [KST21]. The paper further goes on to show that incrementally verifiable computation (IVC) [Val08] can be realised in the random oracle model using a "non-trivial" and non-interactive folding scheme (see Construction 2 and Assumption 1 in [KST21]).

2 *Sangria*: a Folding Scheme for PLONK

Nova builds a folding scheme for the R1CS arithmetization. Here we present a folding scheme for the PLONK arithmetization. We use the same insights as Nova:

- folding is performed by taking a *random linear combination* of the input instance-witness pairs.
- cross terms are absorbed into an *error* (or slack) vector and a *scaling factor*.
- the scheme is made non-trivial by working over *additively homomorphic commitments* to the witness and slack vector.

2.1 Relaxed PLONK Gate Equation

For a scalar $u \in \mathbb{F}$ and error (or slack) vector $\mathbf{e} \in \mathbb{F}^{n+s+1}$ we define the relaxed PLONK gate equation as:

$$C'_{\mathcal{Q},i}(\mathbf{a}, \mathbf{b}, \mathbf{c}, u, \mathbf{e}) := u\left[(\mathbf{q_L})_i \mathbf{a}_i + (\mathbf{q_R})_i \mathbf{b}_i + (\mathbf{q_O})_i \mathbf{c}_i\right] + (\mathbf{q_M})_i \mathbf{a}_i \mathbf{b}_i + u^2(\mathbf{q_C})_i + \mathbf{e}_i \quad (4)$$

Copy constraints in relaxed PLONK are identical to the PLONK copy constraints. A relaxed PLONK trace is represented by the tuple $(\mathbf{a}, \mathbf{b}, \mathbf{c}, u, \mathbf{e})$.

2.2 Folding Scheme for Relaxed PLONK

We show how to build a folding scheme for the relaxed PLONK arithmetization. Let $(\mathbf{X}, \mathbf{W})$ such that $\mathbf{X} := (\mathbf{x_a}, \mathbf{x_b}, \mathbf{x_c}) \in \mathbb{F}^{(n+1)\times 3}$ and $\mathbf{W} := (\mathbf{w_a}, \mathbf{w_b}, \mathbf{w_c}) \in \mathbb{F}^{s\times 3}$ be a satisfying instance-witness pair for the standard PLONK circuit $\mathcal{C}$. Let Com be a hiding and binding additively homomorphic commitment scheme over $\mathbb{F}$ with public parameters pp_W and pp_E to commit to vectors of length s and $n+s+1$ respectively. We define a committed relaxed instance U and committed relaxed witness W as:

$$U := (\mathbf{X}, u, \overline{W_a}, \overline{W_b}, \overline{W_c}, \overline{E}) \quad (5)$$

$$W := (\mathbf{W}, \mathbf{e}, r_a, r_b, r_c, r_e) \quad (6)$$

where $\overline{W_a} = \mathsf{Com}\left(\mathsf{pp}_W, \mathbf{w_a}; r_a\right)$, $\overline{W_b} = \mathsf{Com}\left(\mathsf{pp}_W, \mathbf{w_b}; r_b\right)$, $\overline{W_c} = \mathsf{Com}\left(\mathsf{pp}_W, \mathbf{w_c}; r_c\right)$ and $\overline{E} = \mathsf{Com}\left(\mathsf{pp}_E, \mathbf{e}; r_e\right)$.

Importantly, any PLONK relation can be transformed into a relaxed PLONK relation by setting $u = 1$, $\mathbf{e} = \overrightarrow{0}$ and providing the necessary commitments. Thus the relaxed PLONK arithmetization is NP-complete.

Construction 2.1 (Folding Scheme for Relaxed PLONK)**.** *Consider a finite field $\mathbb{F}$ and a succinct, additively homomorphic, hiding and binding commitment scheme Com over $\mathbb{F}$. Following the notation of [KST21] we define the generator and the encoder as follows:*

- $\mathcal{G}(1^\lambda) \to \mathsf{pp}$*: output size bounds $n, s \in \mathbb{N}$ and commitment parameters pp_W and pp_E for vectors of size s and $n+s+1$ respectively.*
- $\mathcal{K}(\mathsf{pp}, (\mathcal{Q}, \mathcal{S})) \to (\mathsf{pk}, \mathsf{vk})$*: Output* $\mathsf{vk} \leftarrow \perp$ *and* $\mathsf{pk} \leftarrow (\mathsf{pp}, \mathsf{vk}, (\mathcal{Q}, \mathcal{S}))$.

The verifier $\mathcal{V}$ is given the verifier key vk and two committed relaxed PLONK instances, $\left(\mathbf{X}', u', \overline{W'_a}, \overline{W'_b}, \overline{W'_c}, \overline{E'}\right)$ and $\left(\mathbf{X}'', u'', \overline{W''_a}, \overline{W''_b}, \overline{W''_c}, \overline{E''}\right)$. The prover $\mathcal{P}$ is given the prover key pk and both instances with their corresponding witnesses $(\mathbf{W}', \mathbf{e}', r'_a, r'_b, r'_c, r'_e)$ and $(\mathbf{W}'', \mathbf{e}'', r''_a, r''_b, r''_c, r''_e)$.

The Sangria folding scheme proceeds as follows:

1. *$\mathcal{P}$ samples r_t at random and sends $\overline{T} = \mathsf{Com}\left(\mathsf{pp}_E, \mathbf{t}; r_t\right)$ where $\mathbf{t}$ is computed as:*

$$\mathbf{t} := u''(\mathbf{q_L} \circ \mathbf{a}' + \mathbf{q_R} \circ \mathbf{b}' + \mathbf{q_O} \circ \mathbf{c}') + u'(\mathbf{q_L} \circ \mathbf{a}'' + \mathbf{q_R} \circ \mathbf{b}'' + \mathbf{q_O} \circ \mathbf{c}'') \quad (7)$$

$$+ \mathbf{q_M} \circ (\mathbf{a}' \circ \mathbf{b}'' + \mathbf{a}'' \circ \mathbf{b}') \quad (8)$$

$$+ 2ru'u''\mathbf{q_C} \quad (9)$$

2. *$\mathcal{V}$ samples the challenge r at random.*
3. *$\mathcal{P}$ and $\mathcal{V}$ output the folded instance $(\mathbf{X}, u, \overline{W_a}, \overline{W_b}, \overline{W_c}, \overline{E})$ where:*

$$\mathbf{X} \leftarrow \mathbf{X}' + r\mathbf{X}'' \quad (10)$$

$$u \leftarrow u' + ru'' \quad (11)$$

$$\overline{W_a} \leftarrow \overline{W'_a} + r\overline{W''_a} \quad (12)$$

$$\overline{W_b} \leftarrow \overline{W'_b} + r\overline{W''_b} \quad (13)$$

$$\overline{W_c} \leftarrow \overline{W'_c} + r\overline{W''_c} \quad (14)$$

$$\overline{E} \leftarrow \overline{E'} - r\overline{T} + r^2\overline{E''} \quad (15)$$

4. $\mathcal{P}$ outputs the folded witness $(\mathbf{W}, \mathbf{e}, r_a, r_b, r_c, r_e)$ where:

$$\mathbf{W} \leftarrow \mathbf{W}' + r\mathbf{W}'' \tag{16}$$
$$r_a \leftarrow r'_a + r \cdot r''_a \tag{17}$$
$$r_b \leftarrow r'_b + r \cdot r''_b \tag{18}$$
$$r_c \leftarrow r'_c + r \cdot r''_c \tag{19}$$
$$\mathbf{e} \leftarrow \mathbf{e}' - r\mathbf{t} + r^2\mathbf{e}'' \tag{20}$$
$$r_e \leftarrow r'_e - r \cdot r_t + r^2 \cdot r''_e \tag{21}$$

Theorem 2.1. *Construction 2.1 is a public-coin folding scheme for the committed relaxed PLONK arithmetization with perfect completeness, knowledge soundness, and zero-knowledge.*

Proof intuition. Perfect completeness can be shown by following the algebra until establishing that $C'_{\mathcal{Q},i}(\mathbf{a}, \mathbf{b}, \mathbf{c}, u, \mathbf{e}) = C'_{\mathcal{Q},i}(\mathbf{a}', \mathbf{b}', \mathbf{c}', u', \mathbf{e}') + r^2 C'_{\mathcal{Q},i}(\mathbf{a}'', \mathbf{b}'', \mathbf{c}'', u'', \mathbf{e}'')$. We also show that copy constraints are preserved.

We prove knowledge soundness using the same strategy as [KST21]. Specifically, we apply the forking lemma for folding schemes (Lemma 1 in [KST21]) to obtain three transcripts. We then show that the extractor uses all three transcripts to interpolate the original $\mathbf{e}', r'_e$ and $\mathbf{e}'', r''_e$ values, and any two transcripts to interpolate the values $(\mathbf{W}', r'_a, r'_b, r'_c)$ and $(\mathbf{W}'', r''_a, r''_b, r''_c)$. We then show that the interpolated values belong to traces that each satisfy the circuit's gate equalities and copy constraints.

Finally zero-knowledge holds as the prover's messages are hiding commitments and the verifier only sends a public random value. A full proof is presented in Appendix A.

Non-interactive folding scheme. The scheme can be made non-interactive in the random oracle model by applying the Fiat-Shamir transform.

2.3 Degree 2 Custom Gates

In this section we discuss a general strategy to devise a folding scheme for the PLONK arithmetization with custom gates of degree $d_G \leq 2$. We break up the gate equation into sums of monomials of degree 0, 1 and 2. The monomials of degree 1 are multiplied by the scaling factor u, while the cross-terms generated by the degree 2 monomials will get absorbed into a redefined slack vector.

Generic gate equation. Consider a finite field $\mathbb{F}$. Let g be a multivariate polynomial defined over $\mathbb{F}$ of degree $d_G \leq 2$ and define $\mathbf{q_G}$ to be the selector vector for this gate. The gate equation can be written as:

$$G_i(\mathbf{a}, \mathbf{b}, \mathbf{c}) := (\mathbf{q_G})_i \cdot g(\mathbf{a}_i, \mathbf{b}_i, \mathbf{c}_i) \tag{22}$$

The i-th constraint can be written in full as:

$$C_{\mathcal{Q},i}(\mathbf{a}, \mathbf{b}, \mathbf{c}) := (\mathbf{q_L})_i\mathbf{a}_i + (\mathbf{q_R})_i\mathbf{b}_i + (\mathbf{q_O})_i\mathbf{c}_i + (\mathbf{q_M})_i\mathbf{a}_i\mathbf{b}_i + (\mathbf{q_C})_i + (\mathbf{q_G})_i \cdot g(\mathbf{a}_i, \mathbf{b}_i, \mathbf{c}_i) \tag{23}$$

Folding strategy. We write g as a sum of monomials and separate the monomials by their degrees. Let g_C, g_1 and g_2 be the sums of the constant, degree 1 and degree 2 monomials respectively. We can write the relaxed constraint equation as:

$$C'_{\mathcal{Q},i}(\mathbf{a}, \mathbf{b}, \mathbf{c}, u, \mathbf{e}) := u\left[(\mathbf{q_L})_i\mathbf{a}_i + (\mathbf{q_R})_i\mathbf{b}_i + (\mathbf{q_O})_i\mathbf{c}_i + (\mathbf{q_G})_i \cdot g_1(\mathbf{a}_i, \mathbf{b}_i, \mathbf{c}_i)\right] \tag{24}$$
$$+ (\mathbf{q_M})_i\mathbf{a}_i\mathbf{b}_i + (\mathbf{q_G})_i \cdot g_2(\mathbf{a}_i, \mathbf{b}_i, \mathbf{c}_i) + u^2(\mathbf{q_C})_i + u^2(\mathbf{q_G})_i \cdot g_C + \mathbf{e}_i \tag{25}$$

Folding $(\mathbf{a}', \mathbf{b}', \mathbf{c}', u', \mathbf{e}')$ with $(\mathbf{a}'', \mathbf{b}'', \mathbf{c}'', u'', \mathbf{e}'')$ is still performed by taking random linear combinations, however the $\mathbf{t}$ vector must be adjusted to absorb the cross terms that arise from each of the following expressions:

- $(u' + ru'')\big[(\mathbf{q_L})_i(\mathbf{a'}_i + r\mathbf{a''}_i) + (\mathbf{q_R})_i(\mathbf{b'}_i + r\mathbf{b''}_i) + (\mathbf{q_O})_i(\mathbf{c'}_i + r\mathbf{c''}_i)$
  $+ (\mathbf{q_G})_i \cdot g_1((\mathbf{a'}_i + r\mathbf{a''}_i), (\mathbf{b'}_i + r\mathbf{b''}_i), (\mathbf{c'}_i + r\mathbf{c''}_i))\big]$
- $(\mathbf{q_M})_i(\mathbf{a'}_i + r\mathbf{a''}_i)(\mathbf{b'}_i + r\mathbf{b''}_i)$
- $(\mathbf{q_G})_i \cdot g_2\left((\mathbf{a'}_i + r\mathbf{a''}_i), (\mathbf{b'}_i + r\mathbf{b''}_i), (\mathbf{c'}_i + r\mathbf{c''}_i)\right)$
- $(u' + ru'')^2(\mathbf{q_C})_i$
- $(u' + ru'')^2(\mathbf{q_G})_i \cdot g_C$

2.4 Higher Fan-In and Fan-Out

The current scheme can support higher arity circuits as long as the degree of the gate equation is smaller or equal to 2. Each additional gate input or output requires an additional witness column commitment.

3 Future Work

This note establishes a folding scheme for the standard PLONK arithmetization and introduces some customisation features. We conclude by briefly highlighting directions for future (and upcoming) work.

3.1 Succinct IVC using a zkSNARK for *Sangria*

Nova shows that a folding scheme directly implies IVC. However those IVC proofs are neither succinct nor zero-knowledge. To achieve both of these properties, one must devise a zkSNARK for the newly relaxed arithmetization. One possible direction is to convert the *Sangria* trace into a PLONKish trace with an extra witness column for the slack vector. Another direction would be to modify the IOP directly to manage the newly introduced u and $\mathbf{e}$ values.

3.2 Lower Recursion Overhead

In the current construction, the folding verifier works with 1 commitment per witness column. The scheme can also work by flattening the witness matrix $\mathbf{W}$ into a single column vector, thus allowing the verifier to work with a single witness commitment (as in Nova). Doing so requires the reference string pp_W to be three times longer for a circuit with "fan-in 2, fan-out 1" gates. It might also introduce extra checks and commitment openings later in the full IVC scheme given that the standard PLONK IOP uses commitments to each witness column.

3.3 Higher Degree Custom Gates

Higher degree custom gates can be achieved in the "random linear combination" folding strategy. Let d be the *highest* degree of our constraint equation, the overall strategy is the following:

1. Express the non-relaxed constraint equation $C_{\mathcal{Q},i}$ as a sum of monomials.
2. Use powers of the relaxation factor u to make $C_{\mathcal{Q},i}$ into a *homogeneous* degree d polynomial.
3. Add an error term $\mathbf{e}$ to absorb cross-terms. Cross terms will now have powers of r from 1 to $d-1$. We collect them respectively into the vectors $\mathbf{t_1}, \mathbf{t_2}, \ldots, \mathbf{t_{d-1}}$ such that

$$\mathbf{e} = \mathbf{e}' - \sum_{k=1}^{d-1} r^k \mathbf{t_k} + r^d \mathbf{e}'' \tag{26}$$

4. Fold as described in Construction 2.1 with the following modifications:
    - Prover computes and sends commitments to *each* of the $\mathbf{t_k}$ vectors.
    - Compute $\mathbf{e}$ as defined in Equation (26) and apply the corresponding operation in the commitment space.

Costs. This strategy introduces extra work for both the verifier and the prover. To compute the $d-1$ cross-term vectors and their commitments, the prover will perform $\mathcal{O}(d*(n+s))$ field operations and $\mathcal{O}(d*(n+s))$ point additions. Similarly, working in the commitment space, the verifier computes $\mathcal{O}(d)$ point additions. Note that the verifier work remains constant with respect to the security parameter and the circuit size. It does however call for a closer comparison between the folding and split accumulation [BCL+20] approaches to achieve IVC.

Degree 3. A suggestion using the random linear combination strategy for degree 3 gates using the above strategy:

$$u^2\left[(\mathbf{q_L})_i\mathbf{a}_i + (\mathbf{q_R})_i\mathbf{b}_i + (\mathbf{q_O})_i\mathbf{c}_i\right] + u(\mathbf{q_M})_i\mathbf{a}_i\mathbf{b}_i + (\mathbf{q_3})_i\mathbf{a}_i\mathbf{b}_i\mathbf{c}_i + u^3(\mathbf{q_C})_i + \mathbf{e}_i \tag{27}$$

where the error term is appropriately adjusted to absorb cross terms:

$$\mathbf{e} = \mathbf{e}' - r\mathbf{t_1} - r^2\mathbf{t_2} + r^3\mathbf{e}'' \tag{28}$$

$$\mathbf{t_1} = 2u'u''\left[\mathbf{q_L}\circ\mathbf{a} + \mathbf{q_R}\circ\mathbf{b} + \mathbf{q_O}\circ\mathbf{c}\right] + \mathbf{q_M}\circ(u'\mathbf{a}'\circ\mathbf{b}'' + u'\mathbf{a}''\circ\mathbf{b}' + u''\mathbf{a}'\circ\mathbf{b}') \tag{29}$$

$$+ \mathbf{q_3}\circ(\mathbf{a}'\circ\mathbf{b}'\circ\mathbf{c}'' + \mathbf{a}'\circ\mathbf{b}''\circ\mathbf{c}' + \mathbf{a}''\circ\mathbf{b}'\circ\mathbf{c}') \tag{30}$$

$$+ 3(u')^2u''\mathbf{q_C} \tag{31}$$

$$\mathbf{t_2} = \mathbf{q_M}\circ(u''\mathbf{a}''\circ\mathbf{b}' + u''\mathbf{a}'\circ\mathbf{b}'' + u'\mathbf{a}''\circ\mathbf{b}'') \tag{32}$$

$$+ \mathbf{q_3}\circ(\mathbf{a}''\circ\mathbf{b}''\circ\mathbf{c}' + \mathbf{a}''\circ\mathbf{b}'\circ\mathbf{c}'' + \mathbf{a}'\circ\mathbf{b}''\circ\mathbf{c}'') \tag{33}$$

$$+ 3u'(u'')^2\mathbf{q_C} \tag{34}$$

3.4 Lookup Gates

PLONKish arithmetizations differentiate themselves from R1CS in part by their ability to integrate lookup arguments. We are keen to preserve this flexibility by developing folding strategies for lookup-enabled arithmetizations.

Acknowledgements

Thank you to Nat Bunner and Lev Soukhanov for their improvements to the high degree gate folding strategy. We also thank Andrija Novakovic, Lai Ying Tong, Kobi Gurkan and Koh Wei Jie for their helpful inputs and contribution.

References

[BCL+20] Benedikt Bünz, Alessandro Chiesa, William Lin, Pratyush Mishra, and Nicholas Spooner. Proof-carrying data without succinct arguments. Cryptology ePrint Archive, Paper 2020/1618, 2020. https://eprint.iacr.org/2020/1618.

[CBBZ22] Binyi Chen, Benedikt Bünz, Dan Boneh, and Zhenfei Zhang. Hyperplonk: Plonk with linear-time prover and high-degree custom gates. Cryptology ePrint Archive, Paper 2022/1355, 2022. https://eprint.iacr.org/2022/1355.

[GWC19] Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. Plonk: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. Cryptology ePrint Archive, Paper 2019/953, 2019. https://eprint.iacr.org/2019/953.

[KST21] Abhiram Kothapalli, Srinath Setty, and Ioanna Tzialla. Nova: Recursive zero-knowledge arguments from folding schemes. Cryptology ePrint Archive, Paper 2021/370, 2021. https://eprint.iacr.org/2021/370.

[Val08] Paul Valiant. Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In *Theory of Cryptography: Fifth Theory of Cryptography Conference, TCC 2008, New York, USA, March 19-21, 2008. Proceedings 5*, pages 1–18. Springer, 2008.

A Proof of Theorem 2.1

We will show that *Sangria* upholds each of the correctness, knowledge soundness and zero-knowledge properties for folding schemes as required by Definition 6 of [KST21].

Some notation. Recall that the *Sangria* gate equation is expressed as:

$$C'_{\mathcal{Q},i}(\mathbf{a},\mathbf{b},\mathbf{c},u,\mathbf{e}) := u\left[(\mathbf{q_L})_i\mathbf{a}_i + (\mathbf{q_R})_i\mathbf{b}_i + (\mathbf{q_O})_i\mathbf{c}_i\right] + (\mathbf{q_M})_i\mathbf{a}_i\mathbf{b}_i + u^2(\mathbf{q_C})_i + \mathbf{e}_i \quad (35)$$

For ease of notation we define the function $\mathsf{Lin}_{\mathcal{Q}}$ that groups the degree 1 monomials of $C'_{\mathcal{Q}}$:

$$\mathsf{Lin}_{\mathcal{Q},i}(\mathbf{a},\mathbf{b},\mathbf{c}) := (\mathbf{q_L})_i\mathbf{a}_i + (\mathbf{q_R})_i\mathbf{b}_i + (\mathbf{q_O})_i\mathbf{c}_i \quad (36)$$

Notice that $\mathsf{Lin}_{\mathcal{Q},i}(\mathbf{a}+r\mathbf{a}',\mathbf{b}+r\mathbf{b}',\mathbf{c}+r\mathbf{c}') = \mathsf{Lin}_{\mathcal{Q},i}(\mathbf{a},\mathbf{b},\mathbf{c}) + r\mathsf{Lin}_{\mathcal{Q},i}(\mathbf{a}',\mathbf{b}',\mathbf{c}')$ for any scalar r.

Lemma A.1 (Perfect Completeness)**.** *Construction 2.1 is a folding scheme for committed relaxed PLONK with perfect completeness.*

Proof. For any adversarially chosen PLONK circuit $(\mathcal{Q},\mathcal{S})$ and two committed relaxed PLONK instances-witness pairs (U',W') and (U'',W'') we show that the folded instance-witness pair $(U,W) \leftarrow \langle \mathcal{P}(\mathsf{pk},W',W''),\mathcal{V}(\mathsf{vk})\rangle(U',U'')$ also satisfies the circuit (gate constraints, equality constraints and commitment openings).

We denote $\mathbf{M} = (\mathbf{a},\mathbf{b},\mathbf{c})$ the trace defined by the instance-witness pair U,W, u is their scaling factor and $\mathbf{e}$ their slack vector. For vectors $\mathbf{a},\mathbf{b},\mathbf{c}$, we denote $\mathbf{x_a},\mathbf{x_b},\mathbf{x_c}$ their instance parts and $\mathbf{w_a},\mathbf{w_b},\mathbf{w_c}$ their witness parts. We repeat this notation scheme with dashes ($'$) and double dashes ($''$) for the instance-witness pairs U',W' and U'',W''.

(Gate Constraints). Let $\Gamma_i = C'_{\mathcal{Q},i}(\mathbf{a},\mathbf{b},\mathbf{c},u,\mathbf{e})$, we will show that if U',W' and U'',W'' are satisfying traces, then for all $i \in [0,n+s+1]$, $\Gamma_i = 0$. First let us consider the term $u\left[(\mathbf{q_L})_i\mathbf{a}_i + (\mathbf{q_R})_i\mathbf{b}_i + (\mathbf{q_O})_i\mathbf{c}_i\right]$:

$$\begin{aligned}
u\left[(\mathbf{q_L})_i\mathbf{a}_i + (\mathbf{q_R})_i\mathbf{b}_i + (\mathbf{q_O})_i\mathbf{c}_i\right] &= u\left[\mathsf{Lin}_{\mathcal{Q},i}(\mathbf{a}'+r\mathbf{a}'',\mathbf{b}'+r\mathbf{b}'',\mathbf{c}'+r\mathbf{c}'')\right] && (37)\\
&= (u'+ru'')\left[\mathsf{Lin}_{\mathcal{Q},i}(\mathbf{a}',\mathbf{b}',\mathbf{c}') + r\mathsf{Lin}_{\mathcal{Q},i}(\mathbf{a}'',\mathbf{b}'',\mathbf{c}'')\right] && (38)\\
&= u'\mathsf{Lin}_{\mathcal{Q},i}(\mathbf{a}',\mathbf{b}',\mathbf{c}') + r^2u''\mathsf{Lin}_{\mathcal{Q},i}(\mathbf{a}'',\mathbf{b}'',\mathbf{c}'') && (39)\\
&\quad + r\left[u''\mathsf{Lin}_{\mathcal{Q},i}(\mathbf{a}',\mathbf{b}',\mathbf{c}') + u'\mathsf{Lin}_{\mathcal{Q},i}(\mathbf{a}'',\mathbf{b}'',\mathbf{c}'')\right] && (40)\\
&= u'\mathsf{Lin}_{\mathcal{Q},i}(\mathbf{a}',\mathbf{b}',\mathbf{c}') + r^2u''\mathsf{Lin}_{\mathcal{Q},i}(\mathbf{a}'',\mathbf{b}'',\mathbf{c}'') + r\mathbf{t_L}_i && (41)
\end{aligned}$$

where $\mathbf{t_L}_i = u''\mathsf{Lin}_{\mathcal{Q},i}(\mathbf{a}',\mathbf{b}',\mathbf{c}') + u'\mathsf{Lin}_{\mathcal{Q},i}(\mathbf{a}'',\mathbf{b}'',\mathbf{c}'')$. In vector notation:

$$\mathbf{t_L} = u''(\mathbf{q_L}\circ\mathbf{a}' + \mathbf{q_R}\circ\mathbf{b}' + \mathbf{q_O}\circ\mathbf{c}') + u'(\mathbf{q_L}\circ\mathbf{a}'' + \mathbf{q_R}\circ\mathbf{b}'' + \mathbf{q_O}\circ\mathbf{c}'')$$

Similarly, let us consider the term $(\mathbf{q_M})_i\mathbf{a}_i\mathbf{b}_i$:

$$\begin{aligned}
(\mathbf{q_M})_i\mathbf{a}_i\mathbf{b}_i &= (\mathbf{q_M})_i(\mathbf{a}'_i + r\mathbf{a}''_i)(\mathbf{b}'_i + r\mathbf{b}''_i) && (42)\\
&= (\mathbf{q_M})_i\mathbf{a}'_i\mathbf{b}'_i + r\left[(\mathbf{q_M})_i\mathbf{a}'_i\mathbf{b}''_i + (\mathbf{q_M})_i\mathbf{a}''_i\mathbf{b}'_i\right] + r^2(\mathbf{q_M})_i\mathbf{a}''_i\mathbf{b}''_i && (43)\\
&= (\mathbf{q_M})_i\mathbf{a}'_i\mathbf{b}'_i + r\mathbf{t_M}_i + r^2(\mathbf{q_M})_i\mathbf{a}''_i\mathbf{b}''_i && (44)
\end{aligned}$$

where $\mathbf{t_M} = \mathbf{q_M} \circ (\mathbf{a}' \circ \mathbf{b}'' + \mathbf{a}'' \circ \mathbf{b}')$.

Finally let's distribute the term $u^2(\mathbf{q_C})_i$:

$$u^2(\mathbf{q_C})_i = (u' + ru'')^2(\mathbf{q_C})_i \tag{45}$$
$$= u'^2(\mathbf{q_C})_i + r2u'u''(\mathbf{q_C})_i + r^2u''^2(\mathbf{q_C})_i \tag{46}$$
$$= u'^2(\mathbf{q_C})_i + r\mathbf{t_C}_i + r^2u''^2(\mathbf{q_C})_i \tag{47}$$

where $\mathbf{t_C} = 2u'u''(\mathbf{q_C})$.

As per Equation (7), we can write $\mathbf{e}_i = \mathbf{e'}_i + r^2\mathbf{e''}_i - r\mathbf{t_L}_i - r\mathbf{t_M}_i - r\mathbf{t_C}_i$. Substituting the expressions derived above into the constraint equation gives:

$$C'_{\mathcal{Q},i}(\mathbf{a}, \mathbf{b}, \mathbf{c}, u, \mathbf{e}) = u\left[(\mathbf{q_L})_i\mathbf{a}_i + (\mathbf{q_R})_i\mathbf{b}_i + (\mathbf{q_O})_i\mathbf{c}_i\right] + (\mathbf{q_C})_i + (\mathbf{q_M})_i\mathbf{a}_i\mathbf{b}_i + \mathbf{e}_i \tag{48}$$
$$= C'_{\mathcal{Q},i}(\mathbf{a}', \mathbf{b}', \mathbf{c}', u', \mathbf{e}') + r^2C'_{\mathcal{Q},i}(\mathbf{a}'', \mathbf{b}'', \mathbf{c}'', u'', \mathbf{e}'') \tag{49}$$
$$= 0 \tag{50}$$

(Copy Constraints). Without loss of generality, consider the copy constraint $\mathbf{M}_{i,j} = \mathbf{M}_{\alpha,\beta}$ for $i, \alpha \in [0, n+s]$ and $j, \beta \in [0, 2]$. U', W' and U'', W'' are satisfying instance-witness pairs therefore it follows that $\mathbf{M'}_{i,j} = \mathbf{M'}_{\alpha,\beta}$ and $\mathbf{M''}_{i,j} = \mathbf{M''}_{\alpha,\beta}$. It therefore holds that for all $r \in \mathbb{F}$:

$$\mathbf{M'}_{i,j} + r \cdot \mathbf{M''}_{i,j} = \mathbf{M'}_{\alpha,\beta} + r \cdot \mathbf{M''}_{\alpha,\beta} \tag{51}$$
$$\iff \quad \mathbf{M}_{i,j} = \mathbf{M}_{\alpha,\beta} \tag{52}$$

Thus showing that copy constraints are respected.

(Commitment Openings). The final step of our proof is to show that the vectors $\mathbf{w_a}, \mathbf{w_b}, \mathbf{w_c}, \mathbf{e}$ are valid openings for the commitments $\overline{W_a}, \overline{W_b}, \overline{W_c}, \overline{E}$.

For $k \in \{a, b, c\}$, by definition of the folding verifier $\overline{W_k} = \overline{W'_k} + r\overline{W''_k}$. Since Com is an additively homomorphic commitment scheme, it holds that:

$$\overline{W_k} = \overline{W'_k} + r\overline{W''_k} \tag{53}$$
$$= \mathsf{Com}\left(\mathsf{pp}_W, \mathbf{w'_k}; r'_k\right) + r\mathsf{Com}\left(\mathsf{pp}_W, \mathbf{w''_k}; r''_k\right) \tag{54}$$
$$= \mathsf{Com}\left(\mathsf{pp}_W, \mathbf{w'_k} + r\mathbf{w''_k}; r'_k + r \cdot r''_k\right) \tag{55}$$

By definition $\mathbf{w_k} = \mathbf{w'_k} + r\mathbf{w''_k}$ and $r_k = r'_k + r''_k$, therefore $\mathbf{w_k}$ is a valid opening of the commitment $\overline{W_k}$.

The same can be shown for the $\overline{E}$ commitment.

We have now shown that if U', W' and U'', W'' satisfy the circuit $(\mathcal{Q}, \mathcal{S})$, then U, W is also a satisfying instance-witness pair.

□

Lemma A.2 (Knowledge Soundness)**.** *Construction 2.1 is a knowledge sound folding scheme for committed relaxed PLONK if* Com *is a binding commitment.*

Proof. Our proof uses the same strategy as that of Nova [KST21]. Consider a finite field $\mathbb{F}$ and a succinct, additively homomorphic, hiding and binding commitment scheme Com over $\mathbb{F}$. The public parameters consist of size bounds $n, s \in \mathbb{N}$ such that n is the number of public inputs to the circuit and s is the number of gates, and $\mathsf{pp}_W, \mathsf{pp}_E$ commitment parameters for vectors of sizes s and $n + s + 1$ respectively.

For any adversarially chosen PLONK circuit $(\mathcal{Q}, \mathcal{S})$ and two committed relaxed PLONK instances U', U'' we construct an extractor $\mathcal{E}$ that given three accepting transcripts with the same initial commitment $\overline{T}$ outputs the prover's witnesses W', W'' with overwhelming probability.

Let (τ_1, τ_2, τ_3) designate the three transcripts given to $\mathcal{E}$. We designate the verifier randomness of the i-th transcript as $\tau_i.r$. Recall that as per Lemma 1 of [KST21], the transcripts are accompanied by the prover's output folded witness $\tau_i.(\mathbf{W}, \mathbf{e}, r_a, r_b, r_c, r_e)$ where $\mathbf{W} := (\mathbf{w_a}, \mathbf{w_b}, \mathbf{w_c})$ and the output folded instance $\tau_i.(\mathbf{X}, u, \overline{W_a}, \overline{W_b}, \overline{W_c}, \overline{E})$. We abuse notation and write $\tau_i.\mathbf{w_a}$ to denote the witness part of the $\mathbf{a}$-vector corresponding to the i-th transcript.

(Extraction strategy). For each vector $\mathbf{w_k}$ for $k \in \{a, b, c\}$, the extractor interpolates the points $(\tau_1.r, \tau_1.\mathbf{w_k})$, $(\tau_2.r, \tau_2.\mathbf{w_k})$ to retrieve $\mathbf{w_k}'$, $\mathbf{w_k}''$ and points $(\tau_1.r, \tau_1.r_k)$ and $(\tau_2.r, \tau_2.r_k)$ to retrieve r'_k, r''_k such that:

$$\mathbf{w_k}' + \tau_i.r \cdot \mathbf{w_k}'' = \tau_i.\mathbf{w_k} \tag{56}$$

$$r'_k + \tau_i.r \cdot r''_k = \tau_i.r_k \tag{57}$$

for $i \in \{1, 2\}$. Similarly, $\mathcal{E}$ can interpolate the points $(\tau_1.r, \tau_1.\mathbf{e})$, $(\tau_2.r, \tau_2.\mathbf{e})$, $(\tau_3.r, \tau_3.\mathbf{e})$ to retrieve the values $\mathbf{e}'$, $\mathbf{e}''$, $\mathbf{t}$ and points $(\tau_1.r, \tau_1.r_e)$, $(\tau_2.r, \tau_2.r_e)$, $(\tau_3.r, \tau_3.r_t)$ to retrieve r'_e, r''_e, r_t such that:

$$\mathbf{e}' - \tau_i.r \cdot \mathbf{t} + \tau_i.r^2 \mathbf{e}'' = \tau_i.\mathbf{e} \tag{58}$$

$$r'_e - \tau_i.r \cdot r_t + \tau_i.r^2 \cdot r''_e = \tau_i.r_e \tag{59}$$

for $i \in \{1, 2, 3\}$.

(Commitments validity). We must now argue that $W' = (\mathbf{W}', \mathbf{e}', r'_a, r'_b, r'_c, r'_e)$ and $W'' = (\mathbf{W}'', \mathbf{e}'', r''_a, r''_b, r''_c, r''_e)$ satisfy the instances $U' = (\mathbf{X}', u', \overline{W'_a}, \overline{W'_b}, \overline{W'_c}, \overline{E'})$ and $U'' = (\mathbf{X}'', u'', \overline{W''_a}, \overline{W''_b}, \overline{W''_c}, \overline{E''})$ respectively. We first show that for all $k \in \{a, b, c\}$, the retrieved witness vectors $\mathbf{w_k}'$ and $\mathbf{w_k}''$ are valid openings for the commitments $\overline{W'_k}$ and $\overline{W''_k}$ respectively. For $i \in \{1, 2\}$, given that τ_i is an accepting transcript, it holds that $\mathsf{Com}\left(\mathsf{pp}_W, \tau_i.\mathbf{w_k}; \tau_i.r_k\right) = \tau_i.\overline{W_k}$. By construction of the folding scheme:

$$\tau_i.\overline{W_k} = \overline{W'_k} + \tau_i.r\overline{W''_k} \tag{60}$$

Substituting for $\tau_i.\overline{W_k}$, we can get:

$$\begin{aligned}
\overline{W'_k} + \tau_i.r\overline{W''_k} &= \mathsf{Com}\left(\mathsf{pp}_W, \tau_i.\mathbf{w_k}; \tau_i.r_k\right) && (61)\\
&= \mathsf{Com}\left(\mathsf{pp}_W, \mathbf{w_k}' + \tau_i.r \cdot \mathbf{w_k}''; r'_k + \tau_i.r \cdot r''_k\right) && (62)\\
&= \mathsf{Com}\left(\mathsf{pp}_W, \mathbf{w_k}'; r'_k\right) + \tau_i.r\mathsf{Com}\left(\mathsf{pp}_W, \mathbf{w_k}''; r''_k\right) && (63)
\end{aligned}$$

Interpolating with $i \in \{1, 2\}$, we must have that:

$$\mathsf{Com}\left(\mathsf{pp}_W, \mathbf{w_k}'; r'_k\right) = \overline{W'_k} \tag{64}$$

$$\mathsf{Com}\left(\mathsf{pp}_W, \mathbf{w_k}''; r''_k\right) = \overline{W''_k} \tag{65}$$

Similarly, we can show that $\mathbf{e}'$, $\mathbf{e}''$ and $\mathbf{t}$ are valid openings for the commitments $\overline{E'}$, $\overline{E''}$ and $\overline{T}$ respectively. For $i \in \{1, 2, 3\}$, τ_i is an accepting transcript, therefore it holds that $\mathsf{Com}\left(\mathsf{pp}_E, \tau_i.\mathbf{e}; \tau_i.r_e\right) = \tau_i.\overline{E}$. By construction of the folding scheme:

$$\tau_i.\overline{E} = \overline{E'} - \tau_i.r\overline{T} + \tau_i.r^2\overline{E''} \tag{66}$$

Substituting for $\tau_i.\overline{E}$, we get:

$$\overline{E'} - \tau_i.r\overline{T} + \tau_i.r^2\overline{E''} = \mathsf{Com}\left(\mathsf{pp}_E, \tau_i.\mathbf{e};\ \tau_i.r_e\right) \tag{67}$$
$$= \mathsf{Com}(\mathsf{pp}_E, \mathbf{e}' - \tau_i.r\mathbf{t} + \tau_i.r^2\mathbf{e}'';\ \tag{68}$$
$$r'_e - \tau_i.r \cdot r_t + \tau_i.r^2 \cdot r''_e) \tag{69}$$
$$= \mathsf{Com}\left(\mathsf{pp}_E, \mathbf{e}';\ r'_e\right) - \tau_i.r\mathsf{Com}\left(\mathsf{pp}_E, \mathbf{t};\ r_t\right) \tag{70}$$
$$+ \tau_i.r^2\mathsf{Com}\left(\mathsf{pp}_E, \mathbf{e}'';\ r''_e\right) \tag{71}$$

Interpolating with $i \in \{1, 2, 3\}$, we must have that:

$$\mathsf{Com}\left(\mathsf{pp}_E, \mathbf{e}';\ r'_e\right) = \overline{E'} \tag{72}$$
$$\mathsf{Com}\left(\mathsf{pp}_E, \mathbf{t};\ r_t\right) = \overline{T} \tag{73}$$
$$\mathsf{Com}\left(\mathsf{pp}_E, \mathbf{e}'';\ r''_e\right) = \overline{E''} \tag{74}$$

(Constraint satisfaction). We will now show that the extracted witnesses W', W'' and their corresponding instances U', U'' satisfy the circuit defined by $\mathcal{Q}$ and $\mathcal{S}$. For $k \in \{a, b, c\}$, let $\mathbf{k} = \mathbf{w_k}||\mathbf{x_k}$. We start by showing that $(\mathbf{a}', \mathbf{b}', \mathbf{c}', u', \mathbf{e}')$ and $(\mathbf{a}'', \mathbf{b}'', \mathbf{c}'', u'', \mathbf{e}'')$ respectively satisfy the copy constraints defined by $\mathcal{S}$. We then show that they respectively satisfy the gate constraints.

(Copy constraints). Let $\tau_i.\mathbf{M} = (\tau_i.\mathbf{a}, \tau_i.\mathbf{b}, \tau_i.\mathbf{c})$, $\mathbf{M}' = (\mathbf{a}', \mathbf{b}', \mathbf{c}')$ and $\mathbf{M}'' = (\mathbf{a}'', \mathbf{b}'', \mathbf{c}'')$. For $\mathsf{rowA}, \mathsf{rowB} \in [0, n+s]$ and $\mathsf{colA}, \mathsf{colB} \in [0, 2]$, a copy constraint is expressed as:

$$\tau_i.\mathbf{M}_{\mathsf{rowA},\mathsf{colA}} = \tau_i.\mathbf{M}_{\mathsf{rowB},\mathsf{colB}} \tag{75}$$

Substituting with the values from Equation (56), we can write:

$$\mathbf{M}'_{\mathsf{rowA},\mathsf{colA}} + \tau_i.r\mathbf{M}''_{\mathsf{rowA},\mathsf{colA}} = \mathbf{M}'_{\mathsf{rowB},\mathsf{colB}} + \tau_i.r\mathbf{M}''_{\mathsf{rowB},\mathsf{colB}} \tag{76}$$

Interpolating for $i \in \{1, 2\}$, it holds that for all pairs $(\mathsf{rowA}, \mathsf{colA})$, $(\mathsf{rowB}, \mathsf{colB})$:

$$\mathbf{M}'_{\mathsf{rowA},\mathsf{colA}} = \mathbf{M}'_{\mathsf{rowB},\mathsf{colB}} \tag{77}$$
$$\mathbf{M}''_{\mathsf{rowA},\mathsf{colA}} = \mathbf{M}''_{\mathsf{rowB},\mathsf{colB}} \tag{78}$$

(Gate constraints). We will show that for all $j \in [0, n+s]$, $C'_{\mathcal{Q},j}(\mathbf{a}', \mathbf{b}', \mathbf{c}', u', \mathbf{e}') = 0$ and $C'_{\mathcal{Q},j}(\mathbf{a}'', \mathbf{b}'', \mathbf{c}'', u'', \mathbf{e}'') = 0$. First we show that W', W'' satisfy the folding equations for transcript τ_3. Given that τ_3 is an accepting transcript, we know that for all $k \in \{a, b, c\}$, $\mathsf{Com}\left(\mathsf{pp}_W, \tau_3.\mathbf{w_k};\ \tau_3.r_k\right) = \tau_3.\overline{W_k}$. Substituting for τ_3 in Equation (60) yields:

$$\mathsf{Com}\left(\mathsf{pp}_W, \tau_3.\mathbf{w_k};\ \tau_3.r_k\right) = \overline{W'_k} + \tau_3.r\overline{W''_k} \tag{79}$$
$$= \mathsf{Com}\left(\mathsf{pp}_W, \mathbf{w_k}';\ r'_k\right) + \tau_3.r\mathsf{Com}\left(\mathsf{pp}_W, \mathbf{w_k}'';\ r''_k\right) \tag{80}$$
$$= \mathsf{Com}\left(\mathsf{pp}_W, \mathbf{w_k}' + \tau_3.r\mathbf{w_k}'';\ r'_k + \tau_3.r \cdot r''_k\right) \tag{81}$$

Let OPEN_k denote the event that $\tau_3.\mathbf{w_k}$ and $\mathbf{w_k}' + \tau_3.r\mathbf{w_k}''$ are equal, and $\overline{\mathsf{OPEN}}_k$ denote its complement. From the binding property of Com, we know that $\Pr[\overline{\mathsf{OPEN}}_k] = \mathsf{negl}(\lambda)$. Let OPEN denote the event that $\tau_3.\mathbf{W}$ is equal to $\mathbf{W}' + \tau_3.r\mathbf{W}''$ (and $\overline{\mathsf{OPEN}}$ denote its complement):

$$\Pr[\overline{\mathsf{OPEN}}] = \Pr[\overline{\mathsf{OPEN}}_a \wedge \overline{\mathsf{OPEN}}_b \wedge \overline{\mathsf{OPEN}}_c] \tag{82}$$

Therefore it holds that $\Pr[\overline{\mathsf{OPEN}}] \leq max(\Pr[\overline{\mathsf{OPEN}}_a], \Pr[\overline{\mathsf{OPEN}}_b], \Pr[\overline{\mathsf{OPEN}}_c])$ and that $\Pr[\mathsf{OPEN}] \geq 1 - \mathsf{negl}(\lambda)$.

For $i \in \{1, 2\}$, τ_i is an accepting transcript therefore:

$$\Gamma = C'_{\mathcal{Q},i}(\tau_i.\mathbf{a}, \tau_i.\mathbf{b}, \tau_i.\mathbf{c}, \tau_i.u, \tau_i.\mathbf{e}) = 0 \tag{83}$$

Using the equalities from Equation (56) and Equation (58), in the event OPEN it holds that:

$$\begin{aligned}
\Gamma &= u'\mathsf{Lin}_{\mathcal{Q},j}(\mathbf{a}',\mathbf{b}',\mathbf{c}') + (\mathbf{q_M})_j \mathbf{a}'_j \mathbf{b}'_j + \mathbf{e}'_j + u'^2(\mathbf{q_C})_j && (84)\\
&\quad + \tau_i.r\big[u''\mathsf{Lin}_{\mathcal{Q},j}(\mathbf{a}',\mathbf{b}',\mathbf{c}') + u'\mathsf{Lin}_{\mathcal{Q},j}(\mathbf{a}'',\mathbf{b}'',\mathbf{c}'') && (85)\\
&\qquad + (\mathbf{q_M})_j \mathbf{a}'_j \mathbf{b}''_j + (\mathbf{q_M})_j \mathbf{a}''_j \mathbf{b}'_j + 2u'u''(\mathbf{q_C})_j - \mathbf{t}_j\big] && (86)\\
&\quad + \tau_i.r^2\left[\mathsf{Lin}_{\mathcal{Q},j}(\mathbf{a}'',\mathbf{b}'',\mathbf{c}'') + (\mathbf{q_M})_j \mathbf{a}''_j \mathbf{b}''_j + \mathbf{e}''_j + u''^2(\mathbf{q_C})_j\right] && (87)\\
&= C'_{\mathcal{Q},j}(\mathbf{a}',\mathbf{b}',\mathbf{c}',u',\mathbf{e}') && (88)\\
&\quad + \tau_i.r\big[u''\mathsf{Lin}_{\mathcal{Q},j}(\mathbf{a}',\mathbf{b}',\mathbf{c}') + u'\mathsf{Lin}_{\mathcal{Q},j}(\mathbf{a}'',\mathbf{b}'',\mathbf{c}'') && (89)\\
&\qquad + (\mathbf{q_M})_j \mathbf{a}'_j \mathbf{b}''_j + (\mathbf{q_M})_j \mathbf{a}''_j \mathbf{b}'_j + 2u'u''(\mathbf{q_C})_j - \mathbf{t}_j\big] && (90)\\
&\quad + \tau_i.r^2\left[C'_{\mathcal{Q},j}(\mathbf{a}'',\mathbf{b}'',\mathbf{c}'',u'',\mathbf{e}'')\right] && (91)
\end{aligned}$$

Interpolating for the points $(\tau_1.r, 0), (\tau_2.r, 0), (\tau_3.r, 0)$, we get:

$$\begin{aligned}
0 &= C'_{\mathcal{Q},j}(\mathbf{a}',\mathbf{b}',\mathbf{c}',u',\mathbf{e}') && (92)\\
\mathbf{t}_j &= u''\mathsf{Lin}_{\mathcal{Q},j}(\mathbf{a}',\mathbf{b}',\mathbf{c}') + u'\mathsf{Lin}_{\mathcal{Q},j}(\mathbf{a}'',\mathbf{b}'',\mathbf{c}'') && (93)\\
&\quad + (\mathbf{q_M})_j \mathbf{a}'_j \mathbf{b}''_j + (\mathbf{q_M})_j \mathbf{a}''_j \mathbf{b}'_j + 2u'u''(\mathbf{q_C})_j && (94)\\
0 &= C'_{\mathcal{Q},j}(\mathbf{a}'',\mathbf{b}'',\mathbf{c}'',u'',\mathbf{e}'') && (95)
\end{aligned}$$

Thus proving that $(\mathbf{a}',\mathbf{b}',\mathbf{c}',u',\mathbf{e}')$ and $(\mathbf{a}'',\mathbf{b}'',\mathbf{c}'',u'',\mathbf{e}'')$ are satisfying gate assignments in the event OPEN.

(Extractor success). Let $\mathsf{Success}_{\mathcal{E}}$ denote the event in which the extractor $\mathcal{E}$ outputs a pair of valid witnesses W', W'' for the instances U', U''. The above discussion shows that $\Pr[\mathsf{Success}_{\mathcal{E}}] = \Pr[\mathsf{OPEN}]$ and therefore that $\Pr[\mathsf{Success}_{\mathcal{E}}] \geq 1 - \mathsf{negl}(\lambda)$. The existence of this extractor proves the knowledge soundness of Construction 2.1.

□

Lemma A.3 (Zero-knowledge)**.** *Construction 2.1 is a zero-knowledge folding scheme for committed relaxed PLONK if* Com *is a hiding commitment.*

Proof. The prover's only message is hiding commitment. The simulator $\mathcal{S}$ samples a random vector $\mathbf{t} \xleftarrow{\$} \mathbb{F}^{n+s+1}$ and a random scalar $r_T \xleftarrow{\$} \mathbb{F}$. $\mathcal{S}$ can now compute $\overline{T} = \mathsf{Com}\left(\mathsf{pp}_E, \mathbf{t}; r\right)$ and output the transcript $\mathsf{tr} = (\overline{T}, r)$. □