Akita: A High-Performance Lattice-Based Polynomial Commitment Scheme

Quang Dao[1,3], Omid Bodaghi[1], Amirhossein Khajehpour[1], Giuseppe Vitto[1], Mohammadtaghi Badakhshan[1], Markos Georghiades[2], Fengrun Liu[3], Jiapeng Zhang[4], and Justin Thaler[2,5]

[1] LayerZero Labs
[2] a16z crypto research
[3] Carnegie Mellon University
[4] University of Southern California
[5] Georgetown University

Abstract. Lattice-based polynomial commitment schemes (PCSs) promise post-quantum SNARKs with two properties that elliptic curves provide and hash-based schemes, today's deployed post-quantum default, do not: concretely small proofs and commitment time proportional to the number of nonzero entries in the committed polynomial rather than its length. The second property is essential to Twist and Shout (CRYPTO 2026), the fastest known memory-checking arguments and a core component of the Jolt zero-knowledge virtual machine (zkVM): their prover commits to enormous polynomials that are almost entirely zero. Yet despite a wave of recent work, existing lattice-based PCSs achieve at most two of the three properties that deployment demands: small proof size, fast verification, and soundness from standard assumptions such as Module-SIS.

We present Akita, a lattice-based PCS that achieves all three. We improve on the square-root-time verifier of Hachi (ePrint 2026), our direct predecessor, through a new setup offloading technique: the public setup matrices are committed ahead of time, and the verifier's work in processing them is deferred and proved against these commitments. For any fixed $K \geq 2$, this reduces verification time to $\widetilde{O}_{K,\lambda}(N^{1/K})$ while preserving $\widetilde{O}_{K,\lambda}(\log N)$ proof size, $\widetilde{\widetilde{O}}_{K,\lambda}(N)$ prover time, and security from standard Module-SIS. We also optimize every fold from root to tail and iterate the fold to completion. This includes an optimized digit range check, relation-specific ring dimensions and subring challenges, complementary methods for embedding field evaluations and checking ring relations, commitments compressed to 128 bytes each, and exact Euclidean norm checks for tighter Module-SIS parameters.

Beyond the core protocol, Akita provides the capabilities needed for deployment in a zkVM: batched openings of separately committed polynomials, low-communication distributed proving, and an offline planner for selecting secure parameters under configurable cost objectives. We implement Akita in Rust and benchmark it against existing lattice-based and hash-based PCSs. Across these benchmarks, Akita produces proofs of only 58–68 KB, matching Greyhound's when both schemes are calibrated to the same lattice-security target, while verifying 22.1× to 108.7× faster. Akita's prover uses the least memory: beyond storing the polynomial itself, its memory overhead grows sublinearly in the polynomial size. We also integrate Akita into Jolt. For every program size we evaluate, Jolt-with-Akita achieves a 2.2× to 4.2× prover speedup and 2.8× to 8.5× verifier speedup over Jolt-with-Dory, while producing comparable proof sizes, with every proof below 100 KB.

Table of Contents

Akita: A High-Performance Lattice-Based Polynomial Commitment Scheme . . . 1
Quang Dao, Omid Bodaghi, Amirhossein Khajehpour, Giuseppe Vitto, Mohammadtaghi Badakhshan, Markos Georghiades, Fengrun Liu, Jiapeng Zhang, and Justin Thaler

1 Introduction . . . 5
- 1.1 Our Contributions . . . 6
- 1.2 Related Work . . . 9
- 1.3 Organization . . . 11

2 Technical Overview . . . 12
- 2.1 Warm-Up: The Hachi Fold . . . 12
- 2.2 Setup Offloading and the Constant-Root Verifier . . . 13
- 2.3 Optimized Digit Range Check . . . 15
- 2.4 Optimizing the Full Recursion . . . 15
- 2.5 Two Opening Representations . . . 16
- 2.6 Checking Ring Relations with or without Quotients . . . 17
- 2.7 Commitment Compression . . . 19
- 2.8 Bounding Response Coefficients and the Total Squared Norm . . . 19
- 2.9 Batching and Distributed Proving . . . 20
- 2.10 Offline Parameter Selection and Independent Validation . . . 21

3 Preliminaries . . . 21
- 3.1 Cyclotomic Rings, Extension Fields, and Challenge Subrings . . . 21
- 3.2 Sparse and Filtered Folding Challenges . . . 23
- 3.3 Gadget Decomposition and Module-SIS . . . 24
- 3.4 Multilinear Extensions and Sum-Check . . . 24
- 3.5 Equality-Factored Sum-Check . . . 25
- 3.6 Quotient-Lift Ring Switching . . . 26
- 3.7 Tensor Reduction to the Extension Field . . . 27
- 3.8 Polynomial Commitments and Coordinate-Wise Special Soundness . . . 27

4 Setup and Commitments . . . 29
- 4.1 Setting . . . 29
- 4.2 Schedule Topology and Fold Configuration . . . 30
- 4.3 Setup . . . 32
- 4.4 Per-Matrix Ring Dimensions and Challenge Families . . . 33
- 4.5 Commitment . . . 34
- 4.6 Commitments Formed before the Fold . . . 37

5 One Fold for an Opening Batch . . . 38
- 5.1 Inputs and Parameters . . . 38
- 5.2 Shared Tensor Reduction . . . 40
- 5.3 Forming the Partial Opening Evaluations . . . 41
- 5.4 Binding the Partials and Folding the Sources . . . 43
- 5.5 The Witness Passed to the Next Level . . . 44
- 5.6 Interaction and Special Cases . . . 45

6 Norm Checks for the Recursive Witness . . . 47
- 6.1 Digit Range Check . . . 47
- 6.2 Certifying the ℓ_2 Response Norm . . . 49

7 Checking the Ring Relations . . . 52
- 7.1 Opening-Method Evaluation Rows . . . 52
- 7.2 The Common Ring Relation . . . 53
- 7.3 Reducing Ring Equations to Field Identities . . . 53
- 7.4 The Fused Sum-Check . . . 58
- 7.5 Verifier Evaluation of the Batched Row . . . 58

8 Recursive Opening and Terminal Handling . . . 60
- 8.1 Composing the Folds . . . 60
- 8.2 The Terminal Level . . . 60

9 Verifier Offloading . . . 61

9.1 The Setup Claim and Setup-Product Sum-Check . . . 62
9.2 Passing the Setup Opening to the Next Fold . . . 62
9.3 Full Schedules and Admissibility . . . 63
9.4 The Constant-Root Verifier . . . 65
9.5 The Assembled Protocol . . . 66
10 Security of Akita . . . 68
10.1 Challenge Sampling . . . 68
10.2 Completeness . . . 71
10.3 Knowledge Soundness . . . 74
10.4 Schedules Chosen from a Public Family . . . 104
11 Response Chunking for Distributed Proving . . . 106
11.1 The Distributed-Response Obstruction . . . 106
11.2 The Response-Chunked Witness . . . 107
11.3 One Relation over Chunked Witness Columns . . . 107
11.4 Local Images and the Recursive Commitment . . . 108
11.5 Distributed Sum-Check and Verifier Work . . . 109
11.6 Cost, Completeness, and Security . . . 110
12 Offline Planner and Parameter Selection . . . 112
12.1 Parameter-Selection Approaches . . . 113
12.2 Choosing a Complete Schedule . . . 113
12.3 Predicting the Next Fold . . . 114
12.4 Searching Across the Recursion . . . 114
12.5 Validating and Using Schedules . . . 115
12.6 Sum-Check Message Accounting . . . 115
13 Implementation and Evaluation . . . 115
13.1 Implementation and Experimental Methodology . . . 116
13.2 Comparison with Lattice-Based PCSs . . . 116
13.3 Comparison with Hash-Based PCSs . . . 118
13.4 Jolt End-to-End . . . 120
14 Conclusion and Future Work . . . 121
A Notation . . . 127
B Multilinear Evaluation from Structured Factors . . . 128
B.1 From Tensor Factors to Vector Addresses . . . 129
B.2 The General Problem . . . 129
B.3 Finite-State Contraction . . . 130
B.4 Affine Addresses on Intervals . . . 132
B.5 Sharing Equality Work and Combining Families . . . 133
C Evaluating Akita's Public Weights . . . 135
C.1 Quotient-Free Relation Weights . . . 135
C.2 Opening, Range, and Norm Weights . . . 137
C.3 Factored Blocks in the Relation . . . 138
C.4 Shared Setup Views and Offloaded Weights . . . 139
C.5 Verifier Cost of the Batched Row . . . 140
D Deferred Protocol Proofs . . . 140
D.1 Proofs for the Fold . . . 140
D.2 Proofs for the Ring Checks and Setup Offloading . . . 141
E Proofs for Transcript-Tree Extraction . . . 142
E.1 Mixed Transcript Trees . . . 142
E.2 Checkpoint Extraction . . . 143
F Constant-Root Instantiation . . . 147
G SIS Security Estimation . . . 148
H Certifying Operator-Norm Challenge Filters . . . 153
H.1 A Deterministic Fixed-Point Predicate . . . 153
H.2 An Exact Accepted-Support Certificate . . . 154
H.3 Concrete Certificates . . . 155
I Akita in Jolt . . . 156
I.1 Committed Polynomials . . . 156
I.2 From Column Claims to Commitment Openings . . . 156
I.3 One Heterogeneous Akita Proof . . . 157

I.4 Schedules and Setup Offloading . . . 157
I.5 Transcript Grinding in the Backend . . . 158
J Corrections and Security Mismatches in Prior Work . . . 158
J.1 Corrections to Protocol Descriptions . . . 158
J.2 Released Implementations . . . 160
K Response Bounds and Parameter Estimates . . . 163
K.1 Bounds for Honest Responses . . . 164
K.2 From Response Bounds to Completeness . . . 167
K.3 A Response Model for Parameter Selection . . . 168
K.4 A Certified Alternative for Digit Squared Norms . . . 172
K.5 Choosing the Response Digit Depth . . . 173
L Implementation Details . . . 173
L.1 Setup and Commitment Layout . . . 173
L.2 Fold Witness Layout and Encodings . . . 174
L.3 Challenge Sampling . . . 175
M PCS Benchmark Scope and Parameters . . . 175
N Benchmark Methodology and Detailed Measurements . . . 177
N.1 Software and Measurement Procedure . . . 177
N.2 Inputs and Communication Accounting . . . 177
N.3 Jolt Measurement Artifact . . . 178
N.4 Detailed Hash-Based PCS Measurements . . . 178

1 Introduction

Succinct non-interactive arguments of knowledge (SNARKs) are undergoing the same post-quantum transition as the rest of deployed cryptography, away from the elliptic curves that many SNARKs in production today [49,45,28] are based on. Most SNARKs follow a common template: an information-theoretic polynomial interactive oracle proof (PIOP) [32,29] composed with a *polynomial commitment scheme* (PCS) [59]. The migration thus concentrates in the PCS: replacing a curve-based PCS with a post-quantum one makes the entire argument post-quantum.[1]

In practice, this "post-quantum PCS" slot has been filled almost entirely by hash-based schemes: Reed-Solomon-based protocols such as FRI [16], STIR [8], and WHIR [9]; general code-based multilinear commitments such as Ligero [6], Brakedown [48], BaseFold [93], and Ligerito [85]; and the binary-field line of Binius [38,39], Blaze [27], and Bolt [51]. These now underpin deployed proof systems at scale, including Succinct's SP1 [91], Lighter's Plonky2-based exchange rollup [73], and the many provers in the Ethereum Foundation's real-time proving effort, Ethproofs [40,41].

The dominant hash-based PCS paradigm gives up two advantages of elliptic-curve commitments that matter: small proofs and *pay-per-bit* commitment time. On proof size, its concrete proofs commonly reach hundreds of kilobytes: with proven soundness (that is, no reliance on conjectured proximity gaps [7,42]) and practical code rates such as $1/2$ or $1/4$ (avoiding the prover slowdown of lower rates), the smallest proofs (using WHIR) are $O(\lambda^2 \log N + \lambda \log^2 N / \log\log N)$ asymptotically, where N is the polynomial size, and at least 160 KB for non-trivial instances [9,71]. On commitment time, a curve-based commitment applies a linear map directly to the data, $\mathbf{m} \mapsto \sum_i m_i G_i$, for fixed public group elements G_i. Committing to a sparse vector therefore costs only its support, the set of nonzero entries. The dominant Reed–Solomon-and-Merkle construction instead commits to a dense error-corrected codeword, so its standard commitment procedure processes the entire codeword even when the original vector is sparse.[2]

Lattice-based PCSs recover both properties in principle: an Ajtai-style commitment applies a linear map to a small-integer decomposition of the message, retaining pay-per-bit commitment time. Some schemes achieve logarithmic-size opening proofs; among these, some attain concrete proof sizes as small as 50–60 KB [82,80]. A flurry of recent constructions [18,82,80,56,60,61,62,57,55] has pushed the family toward practicality, yet none is ready to be deployed, in a sense we make precise by looking at what our target application demands.[3]

Jolt [10,11] is a zero-knowledge virtual machine (zkVM) that proves correct execution of (64-bit) RISC-V programs, and its Twist and Shout memory-checking arguments [90] commit to *one-hot* polynomials: concatenations of standard basis vectors $\mathbf{e}_j$, one per execution step, with j the memory cell that step touched. Hence, we first desire that commitments and openings exhibit pay-per-bit cost. Second, the opening protocol must verify in tens of milliseconds rather than hundreds of milliseconds or seconds. Third, proof size should be concretely small, tens of kilobytes rather than the hundreds that hash-based schemes pay, and finally, security should rest on standard, well-studied assumptions. Jolt today uses Dory [72], a curve-based PCS that satisfies all four, but is not post-quantum secure; a lattice-based PCS built on Ajtai commitments seems to be the natural replacement.

What, then, prevents Jolt from adopting an existing lattice-based PCS? For existing constructions, the latter three requirements form a *trilemma*: each scheme achieves at most two of the three. Greyhound [82] and Hachi [80] obtain small proofs from standard Module-SIS,[4] but verify in time square-root in the polynomial size. Alternate designs such as RoKoKo [62] and Grand Danois [57] reach polylogarithmic verification

[1] A rigorous proof requires security against quantum adversaries, which means, among other things, that Fiat-Shamir security needs to be in the quantum random oracle model (QROM); we discuss the status of Akita in Section 1.1.

[2] This linear cost is not inherent to every code. Systematic-code constructions such as Brakedown and Bolt [48,51] retain the original message as part of the codeword, and hence could benefit from one-hot or other highly repetitive structure, though the practical speedups remain unclear.

[3] The closest to deployment is the recent LaBRADOR-based zero-knowledge toolkit of Biasioli et al. [19], which extends the LaZer library with a concrete implementation. Its verification time is linear, as in LaBRADOR: on large instances, verification takes more than one second and runs at one-third to one-half of the proving time (see Tables 1 and 2 of [19]).

[4] Module-SIS (along with Module-LWE and their unstructured versions) are among the most well-studied lattice assumptions, with known worst-case-to-average-case reductions [2,64], an extensive history of cryptanalysis, and underlying the security of NIST-standardized ML-DSA [79].

by leaning on stronger, more structured assumption variants (vanishing SIS [34]) that have received far less cryptanalytic attention, and the former also on a heuristic challenge set ensemble. The final set of schemes obtains polylogarithmic verification from standard lattice assumptions [25,3,35], but either gives no concrete parameterization or carries proofs in the megabyte range. This trilemma is only the first obstacle. A deployable PCS must also provide the specific capabilities that a zkVM requires. Namely, Jolt commits to its execution trace, untrusted advice, and preprocessed program data at different stages of the protocol, including offline stages, then opens them together in one proof, each commitment's parameters fixed long before the online commitments are known. Its prover must scale beyond a single machine for large statements; its many interacting parameters must be chosen mechanically and auditably rather than by hand; and its implementation must withstand scrutiny.[5]

This brings us to the central question this paper sets out to answer:

Can we design and implement a lattice-based PCS with concretely small proofs, a fast verifier, and security from standard Module-SIS?

1.1 Our Contributions

We answer this question with Akita, a new lattice-based PCS that satisfies all three properties. We build on Greyhound and Hachi, which achieve small proofs under standard Module-SIS but retain a square-root-time verifier [82,80]. Our new *setup offloading* technique reduces the verifier's asymptotic work to any constant root; with concrete parameter tuning, single-threaded verification takes less than 20 ms even for a 2^{30}-sized polynomial (Theorem 1.1 and Section 13). We also introduce techniques that substantially improve Hachi's central folding reduction, making its range checks and ring-relation checks cheaper and its commitments smaller, and tune these techniques across the recursion to make folding to completion practical. Our evaluation shows that Akita is competitive with existing lattice-based and hash-based PCSs, and when integrated into Jolt, gives 2.2–4.2× faster proving and 2.8–8.5× faster verification, with proofs remaining below 100 KB (Section 13). We prove noninteractive knowledge soundness of Akita in the classical random oracle model (ROM); a proof in the quantum random oracle model (QROM) remains future work (Sections 10 and 14).[6]

Verifier offloading and constant-root verification. We start by isolating the two contributions to Hachi's verifier cost, and explain how we overcome the square-root barrier. For a multilinear polynomial represented by N Boolean evaluations, Hachi divides these evaluations into B blocks; in the first fold, the verifier processes B folding weights and evaluates a public commitment matrix on a block of length roughly N/B. These costs sum to roughly $B + N/B$, minimized at $B \approx \sqrt{N}$; using fewer blocks is offset by a larger matrix evaluation (Section 9.1).

We break this tradeoff by asking the prover to prove the matrix contribution using Akita itself. Since the setup is public and fixed in advance, we can commit to the required setup prefixes during preprocessing.[7] At opening time, an inner-product sum-check reduces the matrix computation to an evaluation of a setup polynomial. We pass this opening to the next fold alongside the recursive-witness opening, so both claims are checked in that fold against their respective commitments (Section 9). We can repeat this process in subsequent folds until offloading no longer provides a benefit, then return to direct matrix evaluation.

We tune the block counts and offloading depth to obtain an asymptotically Kth-root verifier for any constant $K \geq 2$, while preserving quasilinear prover time and polylogarithmic proof size. The recursion schedules keep the combined witness and setup obligations shrinking through the recursion, giving the following guarantee.

[5] A recent security review found critical vulnerabilities in several open-source lattice argument implementations [87]. Appendix J records two issues in public protocol descriptions and several additional mismatches that we found between released implementations and their stated security arguments.

[6] Our Module-SIS parameters are evaluated against quantum lattice attacks, see Appendix G. That cost model is separate from the Fiat–Shamir theorem.

[7] As we explain in Section 4.3, our matrices are sampled as prefixes of a shared flat setup vector of random field elements. This optimization has appeared in prior lattice-based PCS implementations; our security proof explicitly accounts for the shared entries (Section 10).

Table 1: High-level comparison of concretely oriented lattice PCSs. Every scheme has $\widetilde{O}(N)$ prover time. Concrete proof sizes are each work's reported benchmark or estimate, not an apples-to-apples parameter comparison; see Section 1.2 and Appendix J for qualifications.

PCS	Binding assumption	Proof size	Verifier	Artifact
Greyhound [82]	M-SIS	$\widetilde{O}_\lambda(\log N)$; 46–53 KB for $N = 2^{26}$–2^{30}	$\widetilde{O}_\lambda(N^{1/2})$	End-to-end C, AVX-512 only
Hachi [80]	M-SIS	$\widetilde{O}_\lambda(\log N)$; 55 KB for $N = 2^{30}$ (estimate)	$\widetilde{O}_\lambda(N^{1/2})$	Partial Rust, single fold
Jindo [55]	M-SIS	$\widetilde{O}_\lambda(N^{1/3})$; 315–1207 KB for $N = 2^{14}$–2^{20}	$\widetilde{O}_\lambda(N^{1/3})$	End-to-end Go
RoKoKo [62]	vSIS	$\widetilde{O}_\lambda(\log N)$; 115 KB for $N = 2^{26}$–2^{30} (Table 5 in Section 13.2)	$\widetilde{O}_\lambda(\log N)$	End-to-end Rust
Grand Danois [57]	vSIS	$O_\lambda(\log N)$; 80–90 KB for $N = 2^{32}$ (estimate)	$O_\lambda(\log N)$	No implementation
Akita	**M-SIS**	$\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(\log N)$; 61.3–67.3 KB for $N = 2^{22}$–2^{30} (Table 5 in Section 13.2)	$\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N^{1/\kappa_{\mathsf{root}}})$	Production-feature-complete Rust

Theorem 1.1 (Constant-root verifier, informal). *For every fixed integer $K \geq 2$, Akita can be parameterized to open a multilinear polynomial represented by N Boolean evaluations with prover time $\widetilde{O}_{K,\lambda}(N)$, $\widetilde{O}_{K,\lambda}(\log N)$ proof size, and verifier time $\widetilde{O}_{K,\lambda}(N^{1/K})$. Knowledge soundness holds in the classical random-oracle model, assuming Module-SIS remains hard for the growing parameters specified in Section 9.4.*

The verifier's expanded setup has the same asymptotic size as its running time, in addition to the setup seed and commitments to the offloaded prefixes. The notation hides polylogarithmic factors in N and dependence on K and the security parameter λ. We give an explicit family of schedules realizing this tradeoff in Section 9.4, including the required growth of the SIS parameters.

For concrete efficiency, we often offload only one or two levels: removing the largest setup scans already saves substantial verifier work, while further offloading may no longer provide any verifier benefit. This trades prover time for verifier time: at our largest input, offloading more than halves verification while adding roughly 47% to opening time, an overhead that vanishes asymptotically. The proof grows by 2.4–4.4 KB; see Sections 2.4 and 13.2 for precise data.

An optimized digit range check. Norm checks are a central component of lattice-based proof systems; existing works either check coefficient-wise (L_∞) or Euclidean (L_2) norm of the recursive witness. Hachi uses sum-check to prove the former, which takes the form of a degree-b vanishing polynomial for a balanced b-element digit alphabet;[8] for L digits, this incurs $O(b^2 L)$ prover work and $\Theta(b \log L)$ proof size. The optimizations we provide reduce the prover time to $O(bL)$, and, with equality-factored sum-checks [50], the range sum-check round communication to $2(\log_2 b - 1)\lceil \log_2 L \rceil + O(1)$ challenge-field elements; see Sections 2.3 and 6.1.

Our first optimization exploits the symmetry of the balanced alphabet: since the digits k and $-(k+1)$ give the same value under $w \mapsto w(w+1)$, we only need to check half as many transformed values. We then apply a GKR-style product-tree reduction [47] to this smaller $(b/2)$-degree predicate, checking layers with branching factors of two or four to give the smallest proof size. Our implementation further accelerates the prover using the small-value techniques of Dao et al. [37].

A complete opening optimized from root to tail. To make Akita efficient, we must carefully optimize all folding steps, which have very different requirements at the root and toward the tail end. For the former, efficiency

[8] Both Hachi and Akita use the negative-biased balanced alphabet $\mathcal{A}_b := \{-b/2, \ldots, b/2-1\}$. For example, three-bit digits have $b = 2^3 = 8$ and range over $\{-4, -3, -2, -1, 0, 1, 2, 3\}$.

is the most important concern; this is where we optimize for the least verifier work through reducing the setup sizes, and possibly doing offloading on this already-small setup. At the end, we remove components that trade off smaller proof size in that fold for a larger witness in the next fold, and adopt tighter L_2 norm checks (in addition to the digit range check above) to reduce the size of the remaining witness parts. These considerations are possible due to the different ways in which we embed evaluation claims, check ring equations, send commitments, and bound responses across the recursion.

First, to embed evaluation claims for the large early witnesses, we pack coefficients directly into the commitment ring and restrict folding challenges to a subring, allowing each block's partial evaluation to occupy fewer coordinates. In later folds, Hachi's trace-based packing permits smaller commitment rings and shorter opening material, at the cost of using Diamond and Posen's tensor reduction [39] to connect the original base-field evaluation to the packed claim. This reduction supplies the binding missing from Hachi's base-field shortcut [80, Section 3.2, p. 14] (Appendix J.1). We also incorporate the trace identity directly into the field relation, removing the ring element Hachi transmits for its trace check (Section 2.5).

Second, we introduce a *quotient-free* method for checking the ring equations over a field, using sum-check over the field itself. The alternative used in Hachi is the *quotient-lifting* technique of Huang, Mao, and Zhang [54], whose tensor-structured matrix weights give cheaper prover and verifier evaluations, but need to supply additional quotient polynomials that become extra witness material for subsequent folds. Our quotient-free method instead incorporates the ring reduction into public weights, using the same linear map that concurrent Grand Danois [57] expresses through a rotation matrix. These weights are concretely more expensive to process (though asymptotically the same), but eliminating the quotient material benefits the tail end, where witness savings are important. We otherwise adopt the ring quotient-lift sum-check unchanged from Hachi at the early folds (Section 2.6).

Third, we reduce the size of the two commitments sent in each fold by *compressing* them to only 128 bytes each. The idea is simple: we start with normal commitments (which is anywhere from 1–8 KB), then repeatedly digit-decompose then re-commit, using binary digits for the smallest final payload; for 128-bit security, two rounds of this is enough (Section 2.7). This closely matches the idea in Orthus [20]; however, we use a more gradual process of compression, compared to their one-step approach, which yields compressed commitments more than twice as large (Appendix J.2), and handle the resulting binary-ness obligation differently, by fusing with the already-present digit range check. We eventually stop compression when the compression binary digits start becoming a non-trivial part of the next witness.

Fourth, we further reduce the witness size by adopting sum-checks that certify an exact bound on the response's total squared norm, combined with possible operator-norm rejection sampling for the ring folding challenges.[9] While both are established ideas and adopted in many prior works, here our contributions are two-fold: we verify exact, not approximate L_2 norm, and we seamlessly integrate this check with the existing sum-check instances for range and ring-relation checks (Section 2.8). When the field is too small for the whole-response bound to fit, we prove smaller inner products between the digits, recover their integer values, and reconstruct the full squared norm over the integers (Section 6.2). Finally, we adopt LaBRADOR's specialized handling of the terminal opening. In particular, we use a single layer of commitment, send the partial opening evaluations instead of committing to them, and send only the folded response last.

Batching commitments formed at different times. We support opening independently formed commitments in one proof, as required by Jolt, which commits to its trace, advice polynomials, and preprocessed data at different stages of execution. This builds on Greyhound's batching [82, §3.2], but extends it to handle these temporally separated cases. The main issue in our setting is the digit alphabet certified by the shared range check. We fix this alphabet before forming the commitments,[10] while allowing the source, carried-digit, and response bases to differ. Their digits must fit the common alphabet, and each commitment must be sized for its full digit-difference bound. We impose the same compatibility condition when batching a setup-prefix opening with the next recursive witness (Sections 4.6 and 5.6).

Response chunking for distributed proving. For huge polynomials that cannot fit on a single machine, such as arise in Jolt proving of large RISC-V programs, Akita can prove across multiple machines, keeping low

[9] Adopting both these levers can often lower the module rank by 1 or 2, compared to only using the coefficient-wise L_∞ norm check, for the concrete tail witnesses that we observe.

[10] For prover efficiency, we choose $b = 8$.

(sub-square-root) communication. The key bottleneck is the computation of the folded response, which requires an all-gather-reduce followed by an all-broadcast between the machines. The solution is to change the protocol itself: let each machine, which we assume is holding some number of consecutive blocks, compute its own *response chunk* and decompose it, then modify the relation checks to recompose and sum up these chunks whenever needed. This avoids the exchange needed to form a global response, at the cost of a larger recursive witness, proof, and verifier workload. We thus do response-chunking only until communication is cheap enough, then switch to a regular fold; the verifier explicitly rejects protocol parameters with too many chunks, which may inflate its needed work.

Offline planning and independent validation. Akita is a complex protocol with many moving parameters, making it a non-trivial task to choose parameters that lead to secure and efficient protocols. While many parameters have clear, Pareto-optimal choices, others may not, wherein increasing one may positively affect some aspects (e.g., proof size or prove time) at the expense of another (e.g., verifier time).

We provide a comprehensive offline planner that does this automatically, given the input polynomial shapes. Our planner searches exhaustively over complete recursive schedules; for each candidate fold, it derives the witness and any setup opening passed to its successor, then compares the cost of finishing the proof. We set the planner to roughly prioritize verifier efficiency (via the minimum setup size as a rough metric), followed by proof size and prover work. A selected schedule, once created, is validated to be secure independently of the search. We provide a separate validator that reconstructs every folding transition and checks the protocol and security conditions. Prover and verifier then reuse the validated schedule (Sections 10 and 12.3).

Implementation and evaluation. We implement Akita in Rust with optimized field and cyclotomic-ring arithmetic, sparse commitments for Jolt's one-hot inputs, batched openings, distributed proving, and independent schedule validation. Our evaluation answers two questions: whether Akita is competitive with existing lattice-based and hash-based PCSs, and whether Jolt integrated with Akita is meaningfully better than the current version of Jolt (with Dory as the PCS). We report on the numbers, including Akita's strengths and current weaknesses in Section 13.

Another standalone implementation we provide is a SIS estimator written in Rust, following the Python / Sage version [4]. It is about three orders of magnitude faster than the latter, which provides the scalability needed to generate a comprehensive table of secure SIS parameters; our planner subsequently uses these cached tables to generate schedules.

1.2 Related Work

From LaBRADOR to Hachi. We build on a line of lattice arguments that predates the first lattice PCS. Bootle, Lyubashevsky, Nguyen, and Seiler developed sublinear lattice arguments without PCPs [26], while Bootle, Chiesa, and Sotiraki brought sum-check into this setting [24]. ALS20 introduced the weak-opening abstraction and the cross-multiplication argument that underlies its binding [13]. LaBRADOR assembled these ingredients into a recursively self-composed argument with a two-tier Ajtai commitment and an explicit norm reset [18]. Its proofs grow logarithmically, with reported sizes of 47–58 KB for R1CS instances with 2^{10}–2^{20} constraints, but its verifier remains linear in the witness size [18]. Our aim is to retain this concrete compactness while reducing the verifier work.

The short ring challenges used in this lineage have a still older provenance. Sparse, small-norm challenges already appear in lattice identification and Fiat–Shamir protocols [75]. Lyubashevsky and Seiler [77] proved the small-element invertibility theorem for partially splitting cyclotomic rings that makes the difference of two such challenges invertible.

Greyhound turned the LaBRADOR machinery into the first concretely efficient lattice PCS [82]. Its split-and-fold evaluation protocol has a square-root verifier, and recursive compression gives a reported 53 KB proof at degree 2^{30}. Hachi moves the verification of a fold from ring arithmetic to field arithmetic using two ingredients [80]. Its trace identity for embedding extension-field inner products into a cyclotomic ring generalizes the trace technique of LNP22 [76]; its reduction of ring relations to field sum-check is the quotient-based ring-switching method of Huang, Mao, and Zhang [54]. Hachi reports a 12.5× verification speedup over Greyhound at comparable proof size, but its concrete protocol performs one Hachi fold before delegating the remaining witness to a Greyhound tail.

Recent extensions of LaBRADOR. The LaZer toolkit makes LaBRADOR useful for witness-private applications by integrating the linear-size zero-knowledge proof of LNP22 and implementing the resulting composition [19,76]. Orthus addresses a different limitation: it adds a reduction for batched lattice relations that achieves square-root verification under standard assumptions [20]. These are substantive advances in privacy and verification, respectively. Akita's verifier is substantially faster than Orthus's, even without setup offloading, and its security also rests on standard Module-SIS.

Lattice PCSs with single-level folding. CELPC [56] obtains square-root proof size and verification with a transparent setup. Jindo [55] combines ideas from CELPC and Greyhound for client-side proving, reaching cube-root communication and verification and reporting an order-of-magnitude improvement over CELPC across its measured metrics. Both schemes fold only once: they avoid further recursive folds, but their proofs are large. Jindo's concrete parameter choices also include a heuristic challenge set.

Asymptotically succinct lattice PCSs. A separate line aims directly for polylogarithmic proof size and verification. Fenzi, Moghaddas, and Nguyen [43] construct a PCS under a ring variant of BASIS; SLAP [3] uses a multi-instance Power-Ring BASIS assumption and gives a further reduction to Module-SIS; and Cini, Malavolta, Nguyen, and Wee [35] give a transparent-setup construction from standard Module-SIS with knowledge soundness against quantum adversaries. These results obtain stronger asymptotics than Greyhound and Hachi, but their concrete proof sizes are much larger at the scales we target. At degree 2^{30}, the respective papers report proofs of 8.3 MB, 767 MB, and 5.17 MB [43,3,35]. We approach the tradeoff from the other direction: we retain the small concrete proofs of the Greyhound and Hachi family, then improve the verifier both concretely and asymptotically.

Verifier-succinct arguments from structured assumptions. RoK, Paper, SISsors [60] develops verifier-succinct reductions of knowledge for exact norm bounds; RoK and Roll [61] adds structured random projections; SALSAA [63] integrates sum-check to obtain linear prover time; and RoKoKo [62] commits the folding cross-terms rather than sending them, thereby supporting large folding arity. This line obtains polylogarithmic verification by giving the public matrices additional algebraic structure, with its strongest concrete results stated under vanishing SIS or related structured assumptions. We do not claim these asymptotics. Instead, we retain ordinary Module-SIS and, thanks to setup offloading, obtain similar (though still somewhat higher) verifier runtime in practice (Section 9).

Concurrent work: Grand Danois. Closest to Akita within the preceding line, Grand Danois [57] concurrently proposes a lattice PCS with polylogarithmic verification under Module-SIS and vanishing SIS. Its current version checks multiplication by a public ring element through a random field projection of the corresponding negacyclic rotation matrix. This produces the same coefficients as our quotient-free transpose-convolution check. Grand Danois uses that direct check throughout a fixed-dimension relation and obtains succinct setup evaluation from its tensor-structured assumption. We identify the tradeoff between this direct check and the quotient-lifting method used in Hachi, and use the direct check only at the tail of Akita. Grand Danois also integrates a Johnson–Lindenstrauss norm proof into the Greyhound relation and estimates roughly 80 KB proofs for 2^{32}-size evaluations, though without any implementation.

Lattice-based folding schemes. A parallel line studies folding for incrementally verifiable computation and proof-carrying data. LatticeFold [21] gave the first folding scheme from Module-SIS; LatticeFold+ [22] replaced its bit-decomposition range proof with an algebraic one; Neo and SuperNeo [83,84] obtained pay-per-bit commitment costs over small fields; Cyclo [46] obtained additive accumulator-norm growth; Symphony [31] used high-arity folding in a lattice SNARK compiler; and ProtogaLattice [14] folded high-degree relations in a constant number of rounds. Akita shares many techniques and security analyses with these works, but is designed as a lattice-based PCS rather than a folding scheme; we view extending Akita's techniques to a more practical folding scheme as exciting future work.

Concurrent work: PikkuFold. Building on the fixed-weight challenges and operator-norm rejection sketched in LaBRADOR [18], PikkuFold [86] gives a rigorous fixed-Hamming-weight base family with pairwise-invertible differences under the exact partial-splitting conditions of LS18, then measures the concrete rejection rates. PikkuFold also proves a concrete modular Johnson–Lindenstrauss theorem; Akita uses no Johnson–Lindenstrauss projection.

Concrete challenge-set instantiations. Low-norm folding challenges must allow extraction from invertible challenge differences. One approach guarantees that every pair of distinct challenges has an invertible difference, as in the LS18 families. Another uses approximate sampling and bounds the probability of a noninvertible difference, allowing rings with greater splitting and faster arithmetic. RoKoKo uses the latter approach: its concrete numerical estimate is heuristic, while its analysis also cites rigorous bounds and the further treatment in Cyclo [62, Section 9.1]. Probabilistic invertibility guarantees and heuristic numerical estimates should therefore be distinguished. PikkuFold also gives rigorous challenge sets with operator-norm rejection sampling, which can be compared to our parallel development in Appendix H. The two works propose different challenge sets for ring dimension $D = 64$; neither dominates the other in every respect.

Choosing parameters and schedules. We briefly compare Akita's parameter selection with that of other lattice-based schemes. The Greyhound implementation adapts to online-computed folded witness norms during proving, while the LaZer LaBRADOR implementation generates a parameter sequence beforehand from public lengths and norm bounds. The RoKoKo implementation instead supplies concrete parameters at fixed sizes; obtaining new schedules seems non-trivial. Our schedule-generation pipeline is thus arguably the most sophisticated: our planner must tune parameters across multiple folds to minimize the cost of setup offloading. See Section 12.1 for details.

Hash-based PCSs. The hash-based line, from FRI [16] through STIR [8], BaseFold [93], Ligerito [85], and WHIR [9] to Binius [38,39], remains the deployed post-quantum default. These schemes are transparent, their verifiers are concretely fast, and their security rests on hash functions. The Greyhound family instead offers $O(\log N)$ proof growth and linear commitments that can exploit sparse or small-valued data. For dense data, the prover comparison between NTT-based Ajtai commitments and FFT-plus-Merkle encodings is ultimately an implementation question, which we address experimentally rather than by asymptotic analogy (Section 13). Akita also requires preprocessing: besides seeds and offloading commitments, a verifier may retain an expanded public-matrix prefix of size $\widetilde{O}_{K,\lambda}(N^{1/K})$ (Section 9).

Security of prior public artifacts. Several paper descriptions and released implementations in this area differ at security-relevant boundaries. We give the details in Appendix J for the Hachi and Grand Danois papers, the truncated-output Ring-SIS maps in LaBRADOR and Greyhound implementations, the earlier RoKoKo implementation errors and their upstream correction, and the corrected parameter issues in Orthus and PikkuFold.

1.3 Organization

A guide to the paper. The paper is modular, and we do not expect every reader to follow it linearly. Section 2 gives the shortest route through the design and its main tradeoffs; readers interested primarily in performance can then jump to Section 13. Readers who want to check the full construction and security can continue from Section 3 through Section 10; implementers will also want Sections 11 and 12 and Appendix I.

The main body. The overview presents the design ideas in order of their role in the result, whereas the main body gives the complete specification in dependency order. After the algebraic tools of Section 3, Section 4 fixes the setup and commitments and Section 5 constructs one fold for an opening batch. Sections 6 and 7 establish its norm and algebraic checks, and Section 8 closes the recursion and handles the terminal level. Section 9 then uses a second input group to authenticate the setup computation recursively, and Section 10 proves completeness and knowledge soundness of the assembled protocol. Section 11 divides the prover's work among machines, and Section 12 selects and validates the complete schedule.

Experiments and appendices. Section 13 reports the implementation and benchmarks, and Section 14 concludes. The appendices serve as reference: Appendix B develops the structured-tensor evaluator, Appendix G documents SIS estimation, Appendix H certifies the operator-norm filters, Appendix I specifies the Jolt interface, and Appendix J records the corrections needed for security-matched comparisons with prior work.

2 Technical Overview

We start from the Hachi fold [80] and explain how Akita changes its verifier work and the witness passed through recursion. We assume standard SNARK terminology, but no familiarity with lattice proof systems. The warm-up supplies the commitment and folding equations; the rest of the overview explains the choices that make their repeated use efficient.

2.1 Warm-Up: The Hachi Fold

Folding a committed opening. We begin with a multilinear polynomial $\widetilde{f}$ over $\mathbb{F}_q$, represented by its $N = 2^\ell$ Boolean evaluations $f(x) := \widetilde{f}(x)$ for $x \in \{0,1\}^\ell$. The prover has committed to these evaluations and claims $\widetilde{f}(\mathbf{r}) = v$, where $\mathbf{r} \in E^\ell$ for an extension field $E/\mathbb{F}_q$. We reduce this claim to an opening of a smaller multilinear polynomial by dividing the Boolean evaluations into B equal-length blocks and keeping a random linear combination of the blocks.

For lattice commitments, the random coefficients require care. The opening is a short integer vector, and multiplying it by an unrestricted field challenge can make its centered coefficients as large as the modulus, which gives no useful Module-SIS bound. Lattice identification and zero-knowledge protocols therefore draw challenges from a large family of short elements [75,77]. Greyhound and Hachi do so in the cyclotomic ring $R_q := \mathbb{F}_q[X]/(X^d + 1)$. The challenge family $\mathcal{C} \subset R_q$ is large enough for soundness, has small coefficient norms, and has invertible differences between distinct challenges (Section 3.2). For challenges $c_i \in \mathcal{C}$ and the short vector $\mathbf{s}_i$ representing block i, the response is

$$\mathbf{z} := \sum_{i \in [B]} c_i \mathbf{s}_i. \tag{1}$$

The fold must also connect the blocks to their commitment, connect their partial evaluations to v, and prove that the response is short enough for the next fold. LaBRADOR supplies the recursive treatment of small-norm ring relations; Greyhound builds the two-tier commitment and PCS relation; Hachi connects extension-field evaluations to that relation and checks it by field sum-check, using the ring switch of Huang, Mao, and Zhang [18,82,80,54].

Representing the evaluation in the ring. After the standard tensor reduction [39, Theorem 3.5] (Section 3.7), we have an extension-field opening; we reuse $\mathbf{r}$ and v for its point and claimed value. Hachi's trace construction packs $E^{d/k}$ into R_q, where $k = [E : \mathbb{F}_q]$, so that field inner products can be read off ring products through a public linear functional $\mathcal{T}_{\mathbf{r}_{\mathsf{pack}}} : R_q \to E$ (Sections 3.1 and 7.1). After packing, write the blocks as $\mathbf{f}_i \in R_q^M$, so the packed vector contains $N_R := BM$ ring elements. With within-block opening weights $\mathbf{a} \in R_q^M$ and block weights $\chi_{\mathsf{blk}}(i) = \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{blk}}, i)$, the partial opening evaluations satisfy

$$e_i := \langle \mathbf{a}, \mathbf{f}_i \rangle, \qquad v_{\mathsf{R}} := \sum_{i \in [B]} \chi_{\mathsf{blk}}(i) e_i, \qquad \mathcal{T}_{\mathbf{r}_{\mathsf{pack}}}(v_{\mathsf{R}}) = v. \tag{2}$$

Hachi sends v_{R} and checks its trace explicitly. The partials $(e_i)_i$ now carry the evaluation claim. Sending all B of them would add B ring elements to the proof, so we bind them with a short commitment instead.

Committing through small digits. For an even base b, the gadget decomposition $\mathbf{G}_{b,n}^{-1}$ writes each coefficient of n ring elements with digits in $\mathcal{A}_b = \{-b/2, \ldots, b/2 - 1\}$, and $\mathbf{G}_{b,n}$ recomposes them with powers of b, so $\mathbf{G}_{b,n}\mathbf{G}_{b,n}^{-1}(\mathbf{x}) = \mathbf{x}$ (Section 3.3). Decompose each block as $\mathbf{s}_i = \mathbf{G}_{b,M}^{-1}(\mathbf{f}_i)$ and apply an inner commitment matrix $\mathbf{A}$ with n_A rows. The resulting images need not be short, so we decompose them in turn and commit to all of them with an outer matrix $\mathbf{B}$:

$$\mathbf{t}_i := \mathbf{A}\mathbf{s}_i, \qquad \hat{\mathbf{t}}_i := \mathbf{G}_{b_1,n_A}^{-1}(\mathbf{t}_i), \qquad \mathbf{u} := \mathbf{B}\hat{\mathbf{t}}, \quad \hat{\mathbf{t}} := (\hat{\mathbf{t}}_i)_{i \in [B]}. \tag{3}$$

The public commitment is $\mathbf{u}$. The first decomposition gives short openings for the input blocks; the second lets one short message bind all their inner images (Section 4.5).

Binding the partial opening evaluations before folding. The partials use the same pattern. The prover decomposes e_i as $\hat{\mathbf{e}}_i = \mathbf{G}_{b_1,1}^{-1}(e_i)$ and sends $\mathbf{v} := \mathbf{D}\hat{\mathbf{e}}$, where $\hat{\mathbf{e}} := (\hat{\mathbf{e}}_i)_{i\in[B]}$, for a third, independent matrix $\mathbf{D}$. The commitment $\mathbf{u}$ is fixed before the opening point is known. The prover sends $\mathbf{v}$ and v_{R} after receiving the point, and only then does the verifier sample $(c_i)_i$ (Section 5).

Checking the same response against both commitments. The fold must preserve two challenge-dependent equations:

$$\mathbf{A}\mathbf{z} = \sum_{i\in[B]} c_i \mathbf{G}_{b_1,n_A}\hat{\mathbf{t}}_i, \tag{4}$$

$$\mathbf{a}^\top \mathbf{G}_{b,M}\mathbf{z} = \sum_{i\in[B]} c_i \mathbf{G}_{b_1,1}\hat{\mathbf{e}}_i. \tag{5}$$

The first ties the response to the committed blocks; the second ties the same response to their partial evaluations. Opening $\mathbf{u}$ and $\mathbf{v}$ and reconstructing v_{R} supply the remaining constraints. A response coefficient can be larger than an input digit, so the response is decomposed again, at a base b_z and a depth that only needs to cover the admitted response interval (Section 5.5); write $\mathbf{z} = \mathbf{G}_{\mathbf{z}}\hat{\mathbf{z}}$ for its recomposition. The private relation witness is $\mathbf{w}_0 = (\hat{\mathbf{z}}, \hat{\mathbf{e}}, \hat{\mathbf{t}})$ (Section 7.2).

Certifying the digit bounds. These equations hold modulo q and X^d+1; they do not show that the digits are small. A sum-check therefore also proves $Q_b(w) := \prod_{a\in\mathcal{A}_b}(w-a) = 0$ for every digit w. Recomposition ties the digits to the values they represent and the range check bounds them, so the next fold receives a certified short witness. This norm reset keeps response growth from accumulating across folds (Section 6.1).

Returning to a smaller opening claim. The relation is defined over R_q, whereas sum-check runs over a field, and substituting $X = \alpha$ does not respect reduction modulo X^d+1. Hachi uses the HMZ ring switch [54]: a private quotient records the multiples of X^d+1 discarded by each ring equation, giving a polynomial identity that can be evaluated at α. Section 2.6 compares this route with Akita's quotient-free alternative. With quotient digits $\hat{\mathbf{r}}$, the next witness is $\mathbf{w}_{\mathsf{next}} := (\hat{\mathbf{z}}, \hat{\mathbf{e}}, \hat{\mathbf{t}}, \hat{\mathbf{r}})$, and the range and relation checks leave one opening of the multilinear polynomial with these Boolean evaluations, a claim of the original type. Up to security and digit factors, the response has length M and the auxiliaries have length B, so $B \approx M \approx \sqrt{N_R}$ gives a square-root reduction. We suppress the fixed-security factor between N and N_R below.

Binding from weak openings. Consider two accepting executions that differ only in c_i, with challenge and response differences Δc_i and $\Delta\mathbf{z}$. Either the recursively authenticated digits give a short $\mathbf{B}$ or $\mathbf{D}$ collision, or subtraction yields $\Delta c_i \mathbf{s}_i = \Delta\mathbf{z}$. The source $\mathbf{s}_i = (\Delta c_i)^{-1}\Delta\mathbf{z}$ satisfies the commitment and evaluation equations, but the inverse of a short ring element need not be short. Extraction bounds only the product $\Delta c_i \mathbf{s}_i$; this is a *weak opening*. ALS20's cross-multiplication argument, adapted to this commitment by Greyhound and used by Hachi, still gives binding [13, Definition 4.2 and Lemma 4.3][82, Lemma 2.11][80, Lemma 7]. Two distinct sources with the same inner image and bounded products $\rho_i\mathbf{s}_i$ and $\rho_i'\mathbf{s}_i'$, for short units ρ_i, ρ_i', give the short nonzero $\mathbf{A}$-kernel vector $\rho_i'(\rho_i\mathbf{s}_i) - \rho_i(\rho_i'\mathbf{s}_i') = \rho_i\rho_i'(\mathbf{s}_i - \mathbf{s}_i')$. Thus $\mathbf{A}$ is sized for this collision, whereas $\mathbf{B}$ and $\mathbf{D}$ bind digit differences directly. Extraction runs backward from the directly checked terminal to the root (Lemma 10.23 and Theorem 10.40).

Allowing different ring dimensions. Akita chooses separate ring dimensions d_A, d_B, d_D for the inner commitment (and hence the folded response), the outer commitment, and the opening commitment, and reduces each ring equation to a field identity before batching (Remark 7.2 and Section 7.3).

2.2 Setup Offloading and the Constant-Root Verifier

The warm-up verifier processes both the folding challenges and the public commitment matrices. We instead prove the setup-dependent computation and pass its remaining opening claim to the next fold, which yields the verifier of Theorem 1.1; below we write κ_{root} for its root parameter K.

Let $\kappa_\alpha := -(\alpha^d + 1)(\mathbf{G}_{b_1,1})_\alpha$. The four setup- and challenge-bearing rows can be displayed as

row	$\hat{\mathbf{z}}$	$\hat{\mathbf{e}}$	$\hat{\mathbf{t}}$	$\hat{\mathbf{r}}$	RHS
B-open	0	0	$\mathbf{B}_\alpha$	κ_α	$\mathbf{u}_\alpha$
D-open	0	$\mathbf{D}_\alpha$	0	κ_α	$\mathbf{v}_\alpha$
$\hat{\mathbf{z}} \leftrightarrow \hat{\mathbf{t}}$	$(\mathbf{A}\mathbf{G}_\mathbf{z})_\alpha$	0	$(-\mathbf{c}^\top \otimes \mathbf{G}_{b_1,n_A})_\alpha$	κ_α	$\mathbf{0}$
$\hat{\mathbf{z}} \leftrightarrow \hat{\mathbf{e}}$	$(-\mathbf{a}^\top \mathbf{G}_{b,M} \mathbf{G}_\mathbf{z})_\alpha$	$(\mathbf{c}^\top \otimes \mathbf{G}_{b_1,1})_\alpha$	0	κ_α	$\mathbf{0}$

factored: $\widetilde{O}_\lambda(\log N)$ challenges: $\widetilde{O}_\lambda(B)$ setup scan: $\widetilde{O}_\lambda(M+B)$

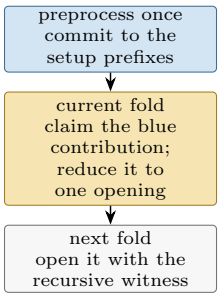


Fig. 1: Verifier cost and setup offloading. Left: verifier cost of each block after substituting $X = \alpha$; row batching does not remove these evaluations. Right: the prover claims the blue contribution, a setup-product sum-check reduces it to one opening, and the next fold checks that opening beside the recursive witness. Offloading removes the setup scan, but not the B-scale challenge evaluation.

The square-root verification cost. At the point $\mathbf{r}_2$ where the relation sum-check ends, the verifier must evaluate the public relation polynomial m_{τ_1}, where τ_1 fixes the combination of relation rows. Write

$$m_{\tau_1}(\mathbf{r}_2) = m_{\mathsf{local}}(\mathbf{r}_2) + \sigma_S, \tag{6}$$

where σ_S is the contribution of the $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{D}$ setup coefficients and m_{local} collects the gadget, challenge, equality, quotient, and compression contributions. Most local weights factor and are cheap to evaluate (Appendix C), but the verifier must still derive all B challenges $c_i(\alpha)$. The $\mathbf{A}$ columns of the relation number about M, and the $\mathbf{B}$ and $\mathbf{D}$ columns about B, so the direct verifier does $\widetilde{O}_\lambda(B+M) = \widetilde{O}_\lambda(B+N/B)$ work, minimized at $B \approx \sqrt{N}$. Figure 1 marks these costs on the relation.

Offloading the matrix evaluations. The prover supplies σ_S and proves it correct. Every matrix reads a prefix of one flat setup sequence $(S_0, S_1, \ldots)$ expanded from the public seed, a layout that also appears in optimized Greyhound implementations [82]. Adding the weights of all uses of each coefficient turns the setup contribution into one inner product, $\sigma_S = \sum_{p<N_{\mathsf{active}}} S_p\, \omega_{\mathsf{setup}}(p)$, over the longest active prefix (Appendix C.4). Lemma 10.20 reduces binding with shared coefficients to the ordinary Module-SIS assumption.

The setup-product sum-check. A degree-two sum-check over the zero-padded prefix $\mathbf{S}^\flat$ reduces σ_S to a claim $s_{\rho_S} = \widetilde{\mathbf{S}^\flat}(\rho_S)$ multiplied by the public factor $\widetilde{\omega}_{\mathsf{setup}}(\rho_S)$, which the verifier evaluates from its description (Appendix C.4). Computing s_{ρ_S} directly would restore the scan. We therefore commit during preprocessing to every setup prefix that a scheduled fold may offload, and fold h passes the setup opening $\mathcal{S}_h = (C_{S,h}, \rho_S, s_{\rho_S})$ beside its recursive-witness opening $\mathcal{W}_{h+1}$:

$$\mathcal{O}_{h+1} = (\mathcal{S}_h, \mathcal{W}_{h+1}). \tag{7}$$

The successor checks both through the multi-instance relation of Section 5. A fold may check the incoming setup opening while producing a new one, so at most one setup opening is pending at any recursive boundary. The successor must check the setup opening like any other opening, so we offload only while this added work is smaller than the removed scan; the final nonterminal fold must check any pending setup opening (Section 9.2).

The constant-root recursion. Fix $\kappa_{\mathsf{root}} \geq 2$ and set $N_\star := \lceil N^{1/\kappa_{\mathsf{root}}} \rceil$. Each of the first $\kappa_{\mathsf{root}} - 1$ folds uses $B_h = \widetilde{\Theta}_{\kappa_{\mathsf{root}},\lambda}(N_\star)$ blocks of length $M_h = \lceil N_h/B_h \rceil$. The verifier still processes the B_h challenges, but offloading removes the longer M_h setup scan. Under the regularity conditions of Section 9.4, the direct evaluations of the small compression maps (Section 2.7) and the other local computations fit within $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N_\star)$ work. The successor holds a response of length about $N_h/N_\star$, $N_\star$-scale auxiliaries, and the setup group, so

$$N_{h+1} = \widetilde{O}_{\kappa_{\mathsf{root}},\lambda}\left(\frac{N_h}{N_\star} + N_\star\right). \tag{8}$$

After $\kappa_{\mathsf{root}} - 1$ such folds the claim has size $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N_\star)$. The witness sizes form a geometric series, so prover time stays quasilinear; a fixed number of wide folds keeps the proof logarithmic; and the first direct setup scan occurs at the $N_\star$ scale, which also bounds the expanded verifier setup. This gives Theorem 1.1; the full accounting appears in Sections 9 and 9.4.

2.3 Optimized Digit Range Check

A larger base b shortens the digit decomposition, but the warm-up's range polynomial Q_b has degree b. Checking it directly widens the sum-check messages and adds prover arithmetic at every fold. A standard GKR product tree [47] lowers the degree of each check, but adds sequential stages and intermediate claims. Before building a tree, we use the particular roots of Q_b.

Pairing the allowed digits. Pair each digit j with $-(j+1)$, for $0 \le j < b/2$. Since $(w-j)(w+j+1) = w(w+1) - j(j+1)$, both digits give the same value of $w(w+1)$, and

$$Q_b(w) = Q_{\mathsf{sq}}\big(w(w+1)\big), \qquad Q_{\mathsf{sq}}(s) := \prod_{j=0}^{b/2-1} \big(s - j(j+1)\big). \tag{9}$$

This symmetry is one reason for the balanced alphabet $\mathcal{A}_b$. The new predicate has degree $b/2$ in s, but s must still be connected to the committed digits.

Authenticating the transformed polynomial. Let $\widetilde{\mathbf{w}}$ be the multilinear polynomial whose Boolean evaluations are the committed digits, and let $\widetilde{\mathbf{s}}$ be the multilinear polynomial with $\widetilde{\mathbf{s}}(x) = \widetilde{\mathbf{w}}(x)(\widetilde{\mathbf{w}}(x)+1)$ on the Boolean cube. The range argument checks Q_{sq} on these values and leaves one claim $s_{\mathsf{claim}} = \widetilde{\mathbf{s}}(\mathbf{r}_{\mathsf{virt}})$. We do not commit to $\widetilde{\mathbf{s}}$. The subsequent relation sum-check proves

$$\sum_{x \in \{0,1\}^{\mu'}} \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{virt}}, x)\widetilde{\mathbf{w}}(x)\big(\widetilde{\mathbf{w}}(x)+1\big) = s_{\mathsf{claim}}.$$

The restriction to Boolean inputs matters: at a general point $\mathbf{r}$, $\widetilde{\mathbf{s}}(\mathbf{r})$ need not equal $\widetilde{\mathbf{w}}(\mathbf{r})(\widetilde{\mathbf{w}}(\mathbf{r})+1)$ (Section 6.1).

Reducing the remaining proof and prover work. For larger bases, a shallow product tree for Q_{sq} combines four factors per layer where possible. Compared with a binary tree, it uses slightly wider round polynomials but fewer stages and intermediate claims (Section 6.1). Most range-tree stages use Gruen's message format, which removes the public equality factor from the round polynomial (Section 3.5 and Table 4; [50]). Following Dao et al. [37], the prover reuses computations because $w(w+1)$ initially takes only $b/2$ values; the Akita Book gives the prover algorithm [65]. A larger base still allows larger digit differences in the binding analysis, so we choose it together with the rest of the recursion.

2.4 Optimizing the Full Recursion

Every object passed to the next witness must be committed, range checked, and folded again, so a choice that helps one fold may leave more work for its successors. As in Hachi's asymptotic analysis, we repeat the fold until direct checking becomes cheaper. The useful choices change as the witness shrinks, and we distinguish three cost regimes below.

A running example. We follow one illustrative schedule through the rest of this overview. The input is a single dense polynomial with 2^{30} arbitrary values in the 32-bit base field $K = \mathbb{F}_q$, occupying 4 GiB as 32-bit words, with one claimed evaluation in the degree-4 extension E/K. This is the field regime and scale of Hachi's concrete discussion [80]. The schedule uses six committed folds, levels 0 through 5, followed by a terminal level T that reveals its response. Its parameters illustrate the comparisons below and are not protocol requirements; Section 12 explains how we generate and validate schedules.

Early folds. The first folds dominate prover and verifier time, so they fold aggressively. Our schedule offloads the root's public-matrix evaluations and checks the resulting setup opening at level 1; later folds evaluate their setup contributions directly. Early folds can also afford auxiliary digits that save public bytes. In the warm-up, the outer matrix $\mathbf{B}$ reads the inner images of all B blocks, so its width grows with B. We instead divide these images into consecutive slices and reuse one narrower matrix on each slice. This reduces the setup footprint and verifier scan, but produces a stack of images, which commitment compression (Section 2.7) keeps from reappearing as proof bytes.

Comparing with direct setup evaluation. Our example offloads a root scan of $5 \cdot 2^{20}$ coefficients through a precommitted prefix padded to 2^{23}. The largest remaining direct scan has 2^{20} coefficients, so expanded verifier matrix storage falls from 22 to 4 MiB. The prover stores 32 instead of 22 MiB of expanded setup and must open the extra prefix, which enlarges the level-1 witness and raises the planner's proof-payload estimate from 63,453 to 66,250 bytes. In the standalone PCS experiment of Section 13, verification falls from 32.2 to 14.1 ms, while opening time grows by 46.7% and the serialized proof by 4.5%. Most of the added prover time comes from the root parameters selected together with offloading, not from opening the setup (Section 13.2).

Recursive tail. For smaller witnesses, fixed proof objects and every value appended to the next witness become a significant fraction of the cost. We therefore stop offloading or compressing as soon as carrying their opening claims costs more than a direct check. Quotient-free relation checks and explicit Euclidean certificates also become attractive when the recursive material they remove exceeds the proof machinery they add. "Early" and "tail" name cost regimes, not fixed level numbers.

Terminal fold. Following LaBRADOR [18, §5.6], the terminal fold omits commitments whose openings would immediately be revealed and does not decompose its response. It fixes its partials $(e_i)_i$ before the challenges, reveals $\mathbf{z} = \sum_i c_i \mathbf{s}_i$, and checks $\mathbf{A}_{\mathsf{term}}\mathbf{z} = \sum_i c_i \mathbf{t}_i$ against the inner images bound by the preceding fold, together with the fold-consistency and scalar-opening equations and the response bound. The auxiliary commitments, quotient digits, range and relation proofs, and next commitment disappear. The last block's $\mathbf{t}_i$ and e_i each appear once, multiplied by an invertible challenge, so the prover sends only a digest of each and the verifier solves for them (Section 8.2).

2.5 Two Opening Representations

Evaluation Trace Diamond and Posen's tensor reduction [39, Theorem 3.5] turns the base-field claim into $\tilde{v} = \theta \bar{f}(\rho)$ at a new point ρ, where $\bar{f}$ packs groups of k base-field Boolean evaluations into E-values and the verifier checks $\theta \neq 0$ (the reduction is the identity for $k = 1$; see Section 3.7). With $m = d_A/k$, Hachi's trace-compatible packing $\psi : E^m \to R_{q,d_A}$, generalizing LNP22 [80,76], places m packed values in each ring element: $F_{i,x} = \psi((\bar{f}_{i,x,u})_u)$ for block i, position x, and packed coordinate u. Evaluating the position coordinates of ρ leaves one ring partial $e_i := \sum_x \widetilde{\mathsf{eq}}(\rho_{\mathsf{pos}}, x) F_{i,x}$ per block. Because these weights embed into the ring, the partial commutes with multiplication by any ring challenge. The packed coordinates are recovered by Hachi's trace pairing

$$\frac{1}{m}\mathrm{Tr}_H\big(\psi(a)\sigma_{-1}(\psi(b))\big) = \langle a, b\rangle, \tag{10}$$

where H is the automorphism subgroup fixing the embedded field E and σ_{-1} sends X to X^{-1}. Pairing with the packed weights $\check{\chi} = \psi((\widetilde{\mathsf{eq}}(\rho_{\mathsf{pack}}, u))_u)$ defines the linear functional $\mathcal{T}(Z) = m^{-1}\mathrm{Tr}_H(Z\sigma_{-1}(\check{\chi}))$, and

$$\theta \sum_i \widetilde{\mathsf{eq}}(\rho_{\mathsf{blk}}, i)\mathcal{T}(e_i) = \tilde{v}. \tag{11}$$

Hachi transmits the ring element v_{R} of (2) and checks its trace separately. We instead prove (11) on the committed opening digits, removing that element (4096 bytes at Hachi's parameters [80, §5.2]) and its ring-switch quotient witness. By linearity of $\mathcal{T}$, this is a degree-two sum-check against the public weights $\omega_{\mathsf{Tr}}(i, \ell, \nu) = \theta\widetilde{\mathsf{eq}}(\rho_{\mathsf{blk}}, i) b_{\mathsf{op}}^{\ell}\mathcal{T}(X^{\nu})$ on digit ℓ of coefficient ν of $\hat{\mathbf{e}}_i$, where b_{op} is the opening-digit base. These

weights factor across their block, digit, and coefficient coordinates, so the verifier evaluates their multilinear extension without scanning the blocks (Appendix C.2). The folded evaluation relation (5) remains ring-valued.

Evaluation trace retains d_A base-field coordinates per partial, but allows full-ring challenges and the small rings useful in later folds.

Direct Coefficient Packing with Subring Challenges When the evaluation-trace partials make up much of the next witness, we shorten them by evaluating some coefficient coordinates before folding, using challenges that preserve the coordinates we retain. This works directly with the base-field polynomial and needs no tensor reduction or trace packing.

Choose powers of two with $d_A = kh_{\mathsf{car}}d_{\mathsf{car}}$, where $k = [E : \mathbb{F}_q]$, and write each coefficient index of the A ring $R := \mathbb{F}_q[X]/(X^{d_A}+1)$ as $\nu = a + kh_{\mathsf{car}}j$ with $a \in [kh_{\mathsf{car}}]$ and $j \in [d_{\mathsf{car}}]$. The source ring element at position x of block i is $F_{i,x} = \sum_{a,j} f_{i,x,a,j}X^{a+kh_{\mathsf{car}}j}$. Summing against the opening weights for x and a, and keeping j as the exponent of a new variable Y, gives one partial per block:

$$e_i(Y) := \sum_{x,a,j} \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{pos}}, x)\widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{pack}}, a) f_{i,x,a,j}Y^j \in E[Y]/(Y^{d_{\mathsf{car}}}+1).$$

It has $kd_{\mathsf{car}} = d_A/h_{\mathsf{car}}$ base-field coordinates, a factor h_{car} fewer than an A-ring element. Multiplication by $X^{kh_{\mathsf{car}}}$ shifts j, including the sign at $X^{d_A} = -1$, and so multiplies the partial by Y. Multiplication by X instead changes the summed index a, and the partial cannot follow it. We therefore draw challenges from the subring

$$S := \mathbb{F}_q[Y]/(Y^{d_{\mathsf{car}}}+1), \qquad \iota : S \hookrightarrow R, \quad \iota(c(Y)) := c(X^{kh_{\mathsf{car}}}),$$

so that folding commutes with taking partials: $\mathcal{L}(\sum_i \iota(c_i)F_i) = \sum_i c_i e_i$, where $\mathcal{L}$ is the weighted sum defining e_i from block F_i. The same challenge acts as c_i here and as $\iota(c_i)$ in the A-ring commitment equations; the embedding preserves its nonzero coefficients, norms, and invertibility. Section 5.3 gives the complete arrangement.

In exchange, d_A must be divisible by kd_{car} and the smaller challenge ring must still contain enough admissible challenges. If this needs more nonzero challenge coefficients, the response bound and the Module-SIS collision radius grow, offsetting part of the saving.

The running example. At the root, the 2^{30} base-field coefficients become 2^{19} A-ring elements at $d_A = 2048$, in $B = 2^{10}$ blocks of $M = 2^9$ positions. With $k = 4$, $d_{\mathsf{car}} = 64$, and $h_{\mathsf{car}} = 8$, the 11 coefficient bits split into a five-bit index a and a six-bit index j. Each partial has $4 \cdot 64 = 256$ base-field coordinates instead of 2048, so the root partials take 1 MiB rather than 8 MiB, and the largest polynomial needs no tensor reduction. Level 1 ($d_A = 1024$, $h_{\mathsf{car}} = 4$) also uses direct packing, for both its recursive-witness and setup groups. The direction reverses at level 2: direct packing would require $4 \cdot 64 = 256 \mid d_A$, whereas evaluation trace permits $d_A = 128$. Holding the block count and digit base fixed, the direct opening would contribute 110,592 digit coordinates to the next witness, twice the 55,296 coordinates of evaluation trace. Table 2 records the opening and ring-relation choices throughout the schedule.

Either representation is available whenever its compatibility conditions hold; the schedules analyzed in this paper fix the choice by level (127), with direct packing at nonterminal levels 0 and 1 and evaluation trace afterwards.

2.6 Checking Ring Relations with or without Quotients

Both opening methods leave equations in a polynomial quotient ring, and a field sum-check must account for reduction modulo X^d+1. For example, $X^3 \cdot X = -1$ in $\mathbb{F}_q[X]/(X^4+1)$, but multiplying the evaluations at $\alpha \in E$ gives α^4. We can either supply a quotient polynomial or reduce each public multiplier before forming its weights on the witness coefficients. Write one ring equation of dimension d as

$$\sum_{c \in \mathsf{Occ}} A_c \circledast_d W_c = Y \qquad \text{in } R_d := K_r[X]/(X^d+1),$$

where K_r is $\mathbb{F}_q$ or, for the direct-packing row, E; the A_c and Y are public; the W_c, indexed by a finite set Occ, are committed witness segments; and $\circledast_d$ denotes multiplication modulo X^d+1.

Table 2: Opening and ring-relation choices in the running 2^{30}-coefficient fp32 schedule. N_h counts the input ring elements before block padding; each of the B_h blocks contains M_h elements, with padding in the last block if needed. The setup row is the additional opening checked at level 1; its $\mathbf{D}$ commitment is shared with the recursive-witness group. The L_2 column records response squared-norm checks; "clear" means direct checking of the revealed response. The terminal still uses tensor reduction and evaluation trace.

level	N_h	M_h	B_h	d_A	d_D	opening	$d_{\mathsf{op},g}$	ring check	L_2 check
0	524,288	512	1,024	2,048	256	direct	256	quotient	–
1	56,149	512	110	1,024	256	direct	256	quotient	–
1 (setup)	8,192	128	64	1,024	256	direct	256	quotient	–
2	54,408	1,024	54	128	128	trace	128	transpose	–
3	7,120	512	14	128	128	trace	128	transpose	–
4	2,704	256	11	128	128	trace	128	transpose	✓
5	1,134	128	9	128	128	trace	128	transpose	✓
T	634	128	5	128	–	trace	128	clear	✓

Checking with a quotient. Following the HMZ ring switch used by Hachi [54,80], the prover supplies Q with

$$\sum_c A_c(X)W_c(X) - Y(X) = (X^d + 1)Q(X),$$

and we check this identity at a random $\alpha \in E$. The weight on each coefficient $w_{c,j}$ of W_c factors as $A_c(\alpha)\alpha^j$, and the coefficient of X^j in Q receives weight $-(\alpha^d + 1)\alpha^j$. This factorization helps both the sum-check prover and the evaluation of public weights for setup offloading. In exchange, the digits of Q join the next witness, where they must be committed, range checked, and folded again.

Including the reduction in the public weights. We can instead reduce the public products before evaluating them, an approach also used concurrently in Grand Danois [57]. For each public multiplier A and coefficient position j, define

$$\kappa^{(d)}_{A,\alpha}(j) := \big(A(X)X^j \bmod (X^d + 1)\big)(\alpha).$$

By linearity, the ring equation becomes

$$\sum_{c \in \mathsf{Occ}} \sum_{j=0}^{d-1} \kappa^{(d)}_{A_c,\alpha}(j) w_{c,j} = Y(\alpha).$$

The reduction signs are already in the weights, so no private quotient is needed. We call this the *transpose-convolution check*, since it moves the public multiplication and evaluation onto the weights of the private coefficients. All d weights for a fixed A can be computed in linear time (Appendix C.1); Section 7.5 explains how the verifier evaluates the resulting public-weight polynomial.

Both methods check the same ring equation. Quotient lifting tests a residual of degree below $2d$, and the direct check tests the reduced residual, of degree below d, so a false row passes at α with probability at most $(2d-1)/|E|$ and $(d-1)/|E|$, respectively. Both use the same subsequent row-batching and sum-check challenges.

Choosing across the recursion. At the early folds, the factored weights of quotient lifting help process the largest witnesses and evaluate offloaded setup weights succinctly. Later, removing a quotient also removes its digits from every subsequent range check, commitment, and fold, which can outweigh the extra work of evaluating the quotient-free relation. In our running schedule, levels 0 and 1 use quotient lifting, levels 2 through 5 use the quotient-free check, and the terminal checks its remaining ring equations directly (Table 2); the planner places this transition by comparing complete schedules.

2.7 Commitment Compression

The public commitment images can occupy a substantial part of the proof even after the witness has become small. An uncompressed fold sends the opening image $\mathbf{v} = \mathbf{D}\hat{\mathbf{e}}$ and the sliced outer image $\mathbf{u}$, with $s_D = n_D d_D$ and $s_B = S_B n_B d_B$ field elements for ranks n_D, n_B and S_B slices. These sizes need not shrink at the same rate as the witness. We reduce them by committing to their binary decompositions, independently for the complete sliced $\mathbf{B}$-image and the unsliced $\mathbf{D}$-image; the recursive relation checks both the recomposition and the new commitment.

Two stages of re-commitment. Let $\mathbf{y}_0 \in \mathbb{F}_q^{s_0}$ be the coefficient vector of either image and $\delta := \lceil \log_2 q \rceil$. We write every coefficient in negative binary, $y = \sum_{i=0}^{\delta-1} 2^i \xi_i \pmod q$ with $\xi_i \in \{-1, 0\}$, and repack the δs_0 digits as a vector $\hat{\mathbf{y}}_1$ over a smaller ring R_{d_1}. A gadget matrix $\mathbf{G}_1$ recomposes the image, and a public matrix $\mathbf{K}_1$ of module rank n_1 commits to the same digits:

$$\mathbf{y}_0 = \mathbf{G}_1 \hat{\mathbf{y}}_1, \qquad \mathbf{y}_1 = \mathbf{K}_1 \hat{\mathbf{y}}_1 \in R_{d_1}^{n_1}. \tag{12}$$

We apply this step again to $\mathbf{y}_1$ and transmit only the final payload $\mathbf{p}$. Both digit vectors join the recursive witness, the relation sum-check checks both links, and the verifier never sees $\mathbf{y}_0$ or $\mathbf{y}_1$. We call the outer chain $\mathbf{F}_1, \mathbf{F}_2$ and the opening chain $\mathbf{H}_1, \mathbf{H}_2$ (Equation (47)). Two legal negative-binary openings differ coefficientwise by at most one, so we size each map as a standalone Module-SIS instance with $\|\Delta\hat{\mathbf{y}}_j\|_\infty \le 1$. The constraint $w(w+1) = 0$ on every compression digit is fused into the existing range-check sum-check, without additional rounds or a higher individual degree (Remark 6.2).

Compression depth and its cost. One small rank-one map cannot securely absorb arbitrary input lengths, so the intermediate image bounds the input to the final map. In our concrete profile, each complete $\mathbf{B}$ stack or shared $\mathbf{D}$ image has at most 8 KiB. Over the 32-bit field, the two rank-one maps use ring dimensions 64 and 32, producing a 256-byte intermediate image and a 128-byte final payload; the 64- and 128-bit profiles also end at 128 bytes (Table 10). The saving adds work to the next fold: for a 32-bit image with s_0 coefficients, the chain adds $32s_0 + 2048$ private binary coordinates, two map relations, and, under quotient lifting, their quotient material. These small maps are evaluated directly even when the main setup scan is offloaded.

Near the root, these additions are small beside the remaining witness; near the tail, they can cost more than sending the uncompressed image. Our running schedule compresses levels 0–4 and sends uncompressed images at level 5, a transition chosen separately from the quotient-free one. Akita permits one transition to uncompressed images, after which compression does not resume; root and setup-prefix commitments remain compressed.

2.8 Bounding Response Coefficients and the Total Squared Norm

The digit-range proof bounds every response coefficient by Z_∞. For a response with N_A base-field coefficients, this only gives $\|\mathbf{z}\|_2^2 \le N_A Z_\infty^2$, which allows all coefficients to approach Z_∞ at once. An honest-response model may predict that this is unlikely, but binding must also hold for adversarial responses.

Certifying the sum of squares. We can therefore also prove $\mathcal{E}_{\mathsf{resp}} := \sum_u z_u^2 \le S_{\max}$. When $S_{\max} < N_A Z_\infty^2$, this can reduce the Module-SIS parameters needed for $\mathbf{A}$, even when both bounds come from the same model: for independent Gaussian coefficients, the coefficient-wise bound can lose a logarithmic factor in squared radius (Appendix K.3).

Checking a squared norm that does not fit in the field. We prove the squared norm from the committed digits of $\mathbf{z}$ and merge it with the final range-check sum-check; the prover sends the claimed norm just before that sum-check. If the largest possible square sum is below q, one field identity certifies the integer equality $\sum_u z_u^2 = S$. In the 32-bit field of our running example, however, the sum can exceed q at the relevant tail dimensions. Write $z_u = \sum_a b^a z_{u,a}$ and partition the coordinates into public segments I_t. We prove the digit inner products $p_{t,a,a'} := \sum_{u \in I_t} z_{u,a} z_{u,a'}$, each small enough to lift uniquely to a centered integer $\bar{p}_{t,a,a'}$, and the verifier reconstructs

$$S = \sum_t \left(\sum_a b^{2a} \bar{p}_{t,a,a} + 2 \sum_{a<a'} b^{a+a'} \bar{p}_{t,a,a'} \right) \tag{13}$$

before checking $S \le S_{\max}$ over the integers. This *digit-expanded norm reconstruction* is detailed in Section 6.2.

Separately controlling challenge multiplication. Binding also depends on how much a challenge can enlarge the response. If κ_1 bounds the coefficient ℓ_1 norm of a folding challenge, Young's inequality gives $\|c\mathbf{z}\|_2 \le \kappa_1\|\mathbf{z}\|_2$. Before invoking a Module-SIS estimator, the two improvements reduce the squared collision radius as

$$64\kappa_1^2 N_A Z_\infty^2 \quad \longrightarrow \quad 64\kappa_1^2 S_{\max} \quad \longrightarrow \quad 64\Gamma^2 S_{\max}. \tag{14}$$

The Euclidean certificate supplies the first arrow. The second is available when accepted challenges satisfy $\|c\mathbf{z}\|_2 \le \Gamma\|\mathbf{z}\|_2$ for every response vector, so Γ bounds how much challenge multiplication stretches a vector. The two gains are independent, and we can use either or both. The coefficient path instead bounds the collision directly in ℓ_∞ as $8\kappa_1 Z_\infty$; the planner compares the resulting Module-SIS ranks, not the norm radii in isolation (Section 6.2 and Appendix H).

Choosing bounds for honest responses. The verifier enforces the selected bounds; separately, an honest prover must be able to meet them. At the root, we use deterministic envelopes or probabilistic bounds over fresh challenge signs for every source satisfying the public conditions, including the effect of any challenge filter (Appendix K.1). For later folds only, a Gaussian-inspired model proposes encoding thresholds and squared-norm bounds. Its predictions are not proved conditioned on the earlier transcript, so completeness for those parameters rests on the explicit premise of Appendix K.3. If the model is optimistic, the honest prover may exhaust its retries; extraction still uses the bounds actually enforced. Certified alternatives are given in Appendices K.2 and K.4. The planner weighs the smaller $\mathbf{A}$ image against the norm-proof and retry costs over the remaining recursion.

2.9 Batching and Distributed Proving

An application such as Jolt opens several commitments, at different points and dimensions. Following Greyhound's batching pattern [82, §3.2], we retain separate responses but share the auxiliary commitment and the arguments that check them. An *opening group* contains one commitment, one point, and its claims at that point; a second point or an independently formed commitment gives another group.

The batch uses one opening representation. Group $g \in [G]$ has $n_{\mathsf{clm},g}$ claims with B_g input blocks each. It keeps its block layout and its own A-ring challenges $c^A_{g,c,i}$, forming

$$\mathbf{z}_g := \sum_{c\in[n_{\mathsf{clm},g}]} \sum_{i\in[B_g]} c^A_{g,c,i}\mathbf{s}_{g,c,i}, \tag{15}$$

which is checked against its own $\mathbf{A}_g$. One unsliced map binds all partial digits, $\mathbf{v} = \mathbf{D}\,\mathsf{concat}(\hat{\mathbf{e}}_0,\dots,\hat{\mathbf{e}}_{G-1})$, and one shared $\mathbf{H}$ chain compresses it when compression is active, while each distinct commitment's outer image has its own $\mathbf{F}$ chain. Fresh scalar weights then combine the opening claims, and one relation argument and one range argument cover the batch. Trace openings first use the shared tensor reduction; direct-packing openings combine their padded claimed values directly, and admissible padding has a nonzero public factor so that it cannot erase a claim (Sections 5.1 and 5.5).

For commitments formed before their batch is known, we fix the common digit alphabet in advance and size every commitment for differences in that full alphabet; otherwise the shared range check could accept digits outside a commitment's binding bound. Each commitment keeps its block length, rings, ranks, slices, and compression path, and the later fold chooses compatible response and opening parameters (Section 4.6). The same mechanism checks a setup-prefix opening beside the recursive witness.

Keeping distributed responses local For distributed proving, machines hold different input blocks. Most commitment computations produce small images that workers can add. The response is harder: for a scheduled partition into N_{chunk} contiguous chunks $\mathcal{I}_j$, a worker forms $\mathbf{z}^{(j)} = \sum_{i\in\mathcal{I}_j} c_i\mathbf{s}_i$, which has the full response length, so forming $\mathbf{z}$ would require exchanging large vectors; and carries prevent the local digit decompositions from summing to the canonical digits of $\mathbf{z}$. We therefore retain and decompose each local response. With equal response digit depths, and excluding shared compression and quotient material, the next witness has length

$$|\mathbf{w}_{\mathsf{chunk}}| = N_{\mathsf{chunk}}|\hat{\mathbf{z}}| + |\hat{\mathbf{e}}| + |\hat{\mathbf{t}}|, \tag{16}$$

so avoiding the exchange adds $(N_{\mathsf{chunk}} - 1)|\hat{\mathbf{z}}|$ coordinates. The relation checks the aggregate effect of these responses by linearity, and the schedule returns to a single response when carrying the extra digits costs more than consolidating them (Section 11). The partition is fixed before the challenges, all chunks share one challenge vector that must make every chunk response admissible, and binding uses the sum of their certified envelopes. We treat the workers and aggregator as one prover; security between those machines requires a separate multiparty protocol (Section 11.6).

2.10 Offline Parameter Selection and Independent Validation

The parameters of one fold cannot be chosen independently of its successors, so we fix them before proving. For each admitted root shape, the planner constructs a complete *schedule*, which the prover and verifier reuse:

$$\mathcal{O}_0 \xrightarrow{\mathsf{Fold}_0} \mathcal{O}_1 \xrightarrow{\mathsf{Fold}_1} \cdots \xrightarrow{\mathsf{Fold}_{L-1}} \mathcal{O}_L \xrightarrow{\mathsf{Terminal}} \mathsf{accept}. \tag{17}$$

Here $\mathcal{O}_h$ is the incoming opening shape. The planner derives each successor, including its auxiliary digits and deferred setup obligation, and compares the cost through the terminal, prioritizing verifier efficiency, then proof size and prover work, keeping the parameters of already-formed commitments fixed. A separate validator checks the selected transitions and complete-schedule conditions; it neither repeats the search nor proves optimality. An opening binds that validated schedule before its first challenge. This separation lets us improve the optimizer while keeping the protocol's acceptance conditions fixed (Section 12).

3 Preliminaries

This section fixes the rings, norms, and challenge families of the commitment scheme; recalls the sum-check, ring-switching, and tensor reductions that turn ring identities into polynomial-evaluation claims; and states the commitment and special-soundness notions used in the security proof. We write $[n] := \{0, \ldots, n-1\}$ and log for the base-2 logarithm.

3.1 Cyclotomic Rings, Extension Fields, and Challenge Subrings

Let q be an odd prime, let d be a power of two, and define the cyclotomic ring $R_q := \mathbb{Z}_q[X]/(X^d + 1)$. An element $w \in R_q$ has coefficient vector $\mathsf{cf}(w) = (w_0, \ldots, w_{d-1}) \in \mathbb{Z}_q^d$. Throughout, α denotes the ring-reduction challenge of both ring-checking routes of Section 7.

Coefficient and multiplication norms. For $w \in R_q$, using centered representatives in $(-q/2, q/2]$, we write

$$\|w\|_\infty := \max_j |w_j \bmod^{\pm} q|, \qquad \|w\|_1 := \sum_j |w_j \bmod^{\pm} q|,$$
$$\|w\|_2 := \Big(\sum_j |w_j \bmod^{\pm} q|^2\Big)^{1/2}.$$

For a vector $\mathbf{w} \in R_q^n$, each norm is taken over the concatenation of all nd centered coefficients.

For $a \in R_q$, let $\mathrm{rot}(a) \in \mathbb{Z}^{d\times d}$ be the negacyclic-convolution matrix of the centered lift of a, and for $\mathbf{v} = (v_1, \ldots, v_n) \in R_q^n$ define

$$\mathrm{rot}(\mathbf{v}) := \begin{pmatrix} \mathrm{rot}(v_1) \\ \vdots \\ \mathrm{rot}(v_n) \end{pmatrix}, \qquad \|\mathbf{v}\|_{\mathsf{mul},2} := \|\mathrm{rot}(\mathbf{v})\|_{2\to 2},$$

the spectral norm of the map $a \mapsto a\mathbf{v}$. For $n = 1$ this is the spectral norm of multiplication by a ring element. Multiplication by a acts block-diagonally on R_q^n, so its operator norm there is still $\|a\|_{\mathsf{mul},2}$, with no factor depending on n. With $\zeta_t := e^{(2t+1)\pi i/d}$, unitary diagonalization of negacyclic convolution gives

$$\|a\|_{\mathsf{mul},2} = \max_{t\in[d]} |a(\zeta_t)|, \qquad \|\mathbf{v}\|_{\mathsf{mul},2} = \max_{t\in[d]} \left(\sum_{j=1}^{n} |v_j(\zeta_t)|^2\right)^{1/2}.$$

The first formula lets the protocol bound the multiplication norm of a challenge by deterministic rejection sampling, provided the accepted family stays large enough for special soundness (Section 3.2). The second lets the verifier compute the multiplication norm of a revealed response directly. The binding proof uses three norm pairings.

Lemma 3.1 (Coefficient-product inequalities). *For every $a \in R_q$ and $\mathbf{v} \in R_q^n$,*

$$\|a\mathbf{v}\|_\infty \le \|a\|_1 \|\mathbf{v}\|_\infty, \tag{18}$$

$$\|a\mathbf{v}\|_2 \le \min\{\|a\|_1\|\mathbf{v}\|_2, \|a\|_{\mathsf{mul},2}\|\mathbf{v}\|_2, \|a\|_2\|\mathbf{v}\|_{\mathsf{mul},2}\}. \tag{19}$$

Proof. Before reduction modulo q, the first bound is the coefficient-wise triangle inequality and the first Euclidean bound is Young's convolution inequality. The second Euclidean bound applies $\mathrm{rot}(a)$ blockwise to the coefficient vector of $\mathbf{v}$. By commutativity we may instead apply $\mathrm{rot}(\mathbf{v})$ to the coefficient vector of a, which gives the third. Centered reduction modulo q cannot increase the absolute value of any coefficient, so all these bounds remain valid in R_q.

Partial splitting and invertibility. Let s_{split} be a power of two with $1 < s_{\mathsf{split}} \le d$. If $q \equiv 2s_{\mathsf{split}}+1 \pmod{4s_{\mathsf{split}}}$, then X^d+1 factors modulo q into exactly s_{split} irreducible polynomials of degree d/s_{split}:

$$X^d + 1 \equiv \prod_{j=1}^{s_{\mathsf{split}}} (X^{d/s_{\mathsf{split}}} - r_j) \pmod{q} \tag{20}$$

for distinct $r_j \in \mathbb{Z}_q^\times$ [77, Corollary 1.2], giving $R_q \cong \prod_{j=1}^{s_{\mathsf{split}}} \mathbb{F}_{q^{d/s_{\mathsf{split}}}}$ by CRT. Then every nonzero $c \in R_q$ with $\|c\|_\infty < q^{1/s_{\mathsf{split}}}/\sqrt{s_{\mathsf{split}}}$ is invertible [77, Corollary 1.2]. Most prior work [18,82,80] uses the minimal splitting $s_{\mathsf{split}} = 2$, i.e. $q \equiv 5 \pmod 8$, with bound $q^{1/2}/\sqrt{2}$; larger s_{split} permits faster NTT-based arithmetic but lowers the bound.

Extension-field embedding. For any $k \ge 1$ dividing d/s_{split}, the field $\mathbb{F}_{q^k}$ embeds into R_q as the subring fixed by a suitable Galois subgroup $H \le \mathrm{Aut}(R_q)$. In the minimal-splitting case, $H = \langle \sigma_{-1}, \sigma_{4k+1} \rangle$ where $\sigma_i : X \mapsto X^i$ ($i \in \mathbb{Z}_{2d}^\times$), and there is an explicit bijection $\psi : (\mathbb{F}_{q^k})^{d/k} \to R_q$ with $\mathrm{Tr}_H(\psi(\mathbf{a}) \cdot \sigma_{-1}(\psi(\mathbf{b}))) = (d/k)\langle \mathbf{a}, \mathbf{b} \rangle$ for all $\mathbf{a}, \mathbf{b} \in (\mathbb{F}_{q^k})^{d/k}$ [80, Theorem 2].

The field $E := \mathbb{F}_{q^k}$ plays one of two roles: the coefficient field of a committed polynomial, or only the challenge and evaluation field of sum-check. The second is common in Akita: we commit to a polynomial through its $\mathbb{F}_q$-valued Boolean evaluations but evaluate it at points of E to make sum-check soundness negligible.

For $k = 1$ the trace construction is the ordinary coefficient embedding.[11] The fixed subring is the coefficient field, ψ is the coefficient-packing map

$$\psi(a_0, \ldots, a_{d-1}) = \sum_{i=0}^{d-1} a_i X^i,$$

and ψ preserves coefficient norms. For $A = \sum_i a_i X^i$ and $B = \sum_i b_i X^i$, the constant coefficient of $A\sigma_{-1}(B)$ is $\sum_i a_i b_i$, so the trace identity is a scaled constant-term extraction.

Ring-subfield basis and packed norms. In the minimal-splitting case, let $k > 1$ be a power-of-two extension degree and set $m := d/(2k)$. Inside R_q, define

$$e_j := X^{jm} + X^{-jm} = X^{jm} - X^{d-jm}, \qquad j \in [1, k),$$

where the second form uses $X^d = -1$. The fixed subfield has the $\mathbb{F}_q$-basis [80, Lemma 4]

$$\boldsymbol{\beta} = (\beta_0, \ldots, \beta_{k-1}) := (1, e_1, \ldots, e_{k-1}),$$

[11] For $k > 1$, placing an element of $\mathbb{F}_{q^k}$ in several coefficient slots of R_q is only $\mathbb{F}_q$-linear and does not preserve multiplication by extension-field points. The trace construction instead uses the multiplicative subfield $R_q^H \simeq \mathbb{F}_{q^k}$.

and the trace embedding is written in these coordinates. The packing map of [80, Theorem 2] combines two banks of m extension elements:

$$\psi(\mathbf{a},\mathbf{b}) = \sum_{t=0}^{m-1} X^t(a_t + X^{d/2}b_t), \qquad a_t = a_{t,0} + \sum_{j=1}^{k-1} a_{t,j}e_j, \tag{21}$$

with the same expansion for b_t. On integer lifts, its packed coefficient vector $\mathbf{c}$ has entries

$$\begin{aligned} c_t &= a_{t,0}, & c_{t+d/2} &= b_{t,0}, \\ c_{t+jm} &= a_{t,j} + b_{t,k-j}, & c_{t+d-jm} &= -a_{t,j} + b_{t,k-j} \quad (1 \le j < k). \end{aligned} \tag{22}$$

The two banks overlap, so packing can increase the coefficient ℓ_∞ and ℓ_1 norms by a factor of at most 2 and the ℓ_2 norm by at most $\sqrt{2}$, since $(A+B)^2 + (-A+B)^2 = 2(A^2+B^2)$ for each pair; centered reduction modulo q only decreases them. We call these the *logical-to-ring factors*. They apply only to a bound stated before ψ. A norm stated on the centered ring coefficients of a folded response $\mathbf{z}$, or on the ring coefficients reconstructed from its digits $\hat{\mathbf{z}}$, is already measured in the ring-coefficient space of Module-SIS and receives no further factor.

3.2 Sparse and Filtered Folding Challenges

A folding step samples challenges from a finite set $\mathcal{C} \subset R_q$ of sparse elements: each challenge has a small random support, fixed magnitudes on that support, and independent random signs. The family is fixed by the ring dimension d and counts $(n_a)_{a\in\mathcal{A}}$, where $\mathcal{A} \subset \mathbb{Z}_{>0}$ is a finite set of allowed magnitudes and $n_a \ge 0$ is the number of support coordinates of absolute value a. We require $1 \le s := \sum_{a\in\mathcal{A}} n_a \le d$ and write

$$\omega := \sum_{a\in\mathcal{A}} a\, n_a, \qquad c_{\max} := \max\{a \in \mathcal{A} : n_a > 0\}.$$

A challenge is drawn in three steps (Appendix L.3 gives an exact sampler):

1. choose a uniform s-subset $S \subseteq \{0,\dots,d-1\}$;
2. choose uniformly a partition of S into disjoint blocks $(S_a)_{a\in\mathcal{A}}$ with $|S_a| = n_a$;
3. for each $a \in \mathcal{A}$ and $j \in S_a$, set $c_j = \varepsilon_j a$ with independent uniform signs $\varepsilon_j \in \{\pm 1\}$, and set $c_j = 0$ for $j \notin S$.

Every member satisfies $\|c\|_\infty = c_{\max}$ and $\|c\|_1 = \omega$, and

$$|\mathcal{C}| = \binom{d}{s}\binom{s}{n_a : a \in \mathcal{A}} 2^s,$$

where $\binom{s}{n_a : a\in\mathcal{A}} = s!/\prod_a n_a!$ is the multinomial coefficient.

A *filter* is a fixed, witness-independent predicate on $\mathcal{C}$. We then call $\mathcal{C}$ the *base family* and the set of its elements that pass the filter the *accepted family* $\mathcal{C}^{\mathsf{acc}} \subseteq \mathcal{C}$; without a filter, $\mathcal{C}^{\mathsf{acc}} = \mathcal{C}$. The verifier samples from $\mathcal{C}^{\mathsf{acc}}$ by rejection: it draws uniform elements of $\mathcal{C}$ and returns the first one that passes the filter, or $\perp$ after a scheduled *draw limit* L (Lemma 10.1). A fold with W challenge coordinates has coordinate-wise special-soundness error $W/|\mathcal{C}^{\mathsf{acc}}|$ (Section 3.8). For extraction, a challenge difference $\bar{c} = c - c'$ satisfies $\|\bar{c}\|_1 \le 2\omega$ and $\|\bar{c}\|_\infty \le 2c_{\max}$ by the triangle inequality. Under the splitting (20), every nonzero such difference is therefore invertible once $2c_{\max} < q^{1/s_{\mathsf{split}}}/\sqrt{s_{\mathsf{split}}}$; the admitted challenge family must satisfy this condition at the selected modulus.

Some Euclidean security profiles filter the family by multiplication norm. For a base family $\mathcal{C}_0$ as above and a public threshold Γ, the accepted family is

$$\mathcal{C}_\Gamma^{\mathsf{acc}} := \{c \in \mathcal{C}_0 : \mathsf{Acc}_\Gamma(c) = 1\}, \tag{23}$$

where Acc_Γ is a deterministic, witness-independent predicate that accepts only challenges with $\|c\|_{\mathsf{mul},2} \le \Gamma$. Rejection sampling from $\mathcal{C}_0$ makes the output uniform over $\mathcal{C}_\Gamma^{\mathsf{acc}}$. Appendix H defines the predicate, proves this norm bound (Theorem H.1), and certifies the lower bound on $|\mathcal{C}_\Gamma^{\mathsf{acc}}|$ that enters the soundness error.

3.3 Gadget Decomposition and Module-SIS

Fix a power-of-two decomposition base $b > 1$ and let $\delta := \lceil \log_b q \rceil$. The *gadget matrix* $\mathbf{G}_{b,n} := I_n \otimes (1\ b\ \cdots\ b^{\delta-1}) \in R_q^{n \times n\delta}$ reconstructs a vector from its digit representation: $\mathbf{G}_{b,n} \cdot \mathbf{s} = \sum_{i=0}^{\delta-1} b^i \mathbf{s}_i$. The decomposition map $\mathbf{G}_{b,n}^{-1} : R_q^n \to R_q^{n\delta}$ is a fixed right inverse of $\mathbf{G}_{b,n}$ that writes each coefficient in balanced base-b digits, drawn from the alphabet

$$\mathcal{A}_b := \{-b/2, \ldots, b/2 - 1\}.$$

Because $\mathcal{A}_b$ is asymmetric, so is the set of integers $\sum_{i<k} d_i b^i$ with $d_i \in \mathcal{A}_b$: it is the interval $[-M_k, T_k]$ of exactly b^k integers, where

$$T_k := \left(\frac{b}{2} - 1\right) \frac{b^k - 1}{b - 1}, \qquad M_k := \frac{b}{2} \cdot \frac{b^k - 1}{b - 1}.$$

For full-field decompositions we take $k = \delta$ and lift a canonical residue $x \in [0, q)$ to the integer

$$\tilde{x} := \begin{cases} x & \text{if } x \le T, \\ x - q & \text{if } x > T, \end{cases} \qquad T := \min(T_k, \lfloor q/2 \rfloor).$$

Then $\tilde{x} \in [-M_k, T_k]$, so no extra digit is needed. Different protocol components may use different bases, such as the commitment base b and the opening base b_1; we keep the base explicit in $\mathbf{G}_{b,n}$ whenever more than one is in play.

Definition 3.2 (Module-SIS). *For $p \in \{2, \infty\}$ and a bound $\beta > 0$, the $\mathsf{MSIS}_{q,d}^{(p)}(n, m, \beta)$ problem asks, given a uniform matrix $M \in R_q^{n \times m}$, to find a nonzero $\mathbf{z} \in R_q^m$ with $M\mathbf{z} = \mathbf{0}$ and $\|\mathbf{z}\|_p \le \beta$. We write $\mathsf{MSIS}_{q,d}$ for the case $p = \infty$.*

Akita uses the coefficient-ℓ_∞ problem for digit-alphabet collisions. An inner-commitment collision may instead use the Euclidean problem when the verifier certifies the required response norm, through one of the three pairings of Lemma 3.1. Appendix G documents the hardness targets and the offline estimators.

3.4 Multilinear Extensions and Sum-Check

Let $\mathbb{F}$ be a finite field. For a function $f : \{0,1\}^\mu \to \mathbb{F}$, its *multilinear extension* (MLE) is

$$\widetilde{f}(X_1, \ldots, X_\mu) := \sum_{i \in \{0,1\}^\mu} f(i) \cdot \widetilde{\mathsf{eq}}(i, X),$$

$$\widetilde{\mathsf{eq}}(i, X) := \prod_{j=1}^{\mu} \big((1 - i_j)(1 - X_j) + i_j X_j\big).$$

When f is given as a vector $\mathbf{f} \in \mathbb{F}^{2^\mu}$ we identify $\widetilde{\mathbf{f}}$ with the MLE of $i \mapsto f_i$.

Theorem 3.3 (Sum-check soundness). *The sum-check protocol (Figure 2) satisfies the following:*

1. Interactive soundness. *If $T \neq \sum_{x \in \{0,1\}^\mu} g(x)$, then a cheating prover causes the verifier to accept with probability at most $\mu\ell/|\mathbb{F}|$ [74].*
2. $(\ell + 1)$-special soundness. *Fix the prefix and the prover's round-j message. If $\ell + 1$ distinct challenges $r_j^{(1)}, \ldots, r_j^{(\ell+1)}$ have continuations certifying the corresponding honest residual sums, then interpolation identifies the round polynomial with the honest degree-at-most-ℓ polynomial s_j [80, Lemma 9]. In a special-sound transcript tree, those residual sums are certified by backward extraction from the final checks. Composing all μ rounds, the sum-check protocol is $(1, \ldots, 1)$-coordinate-wise $(\ell + 1, \ldots, \ell + 1)$-special sound (Definition 3.9), with knowledge error $\mu\ell/|\mathbb{F}|$.*

Claim: $T = \sum_{x\in\{0,1\}^\mu} g(x)$, where $g : \mathbb{F}^\mu \to \mathbb{F}$ has individual degree $\le \ell$.

Verifier state: running claim $T_1 := T$.

For $j = 1, \dots, \mu$:
1. $\mathcal{P} \to \mathcal{V}$: the round polynomial $s_j(X_j) := \sum_{x_{j+1},\dots,x_\mu\in\{0,1\}} g(r_1, \dots, r_{j-1}, X_j, x_{j+1}, \dots, x_\mu)$ of degree $\le \ell$, *with its linear coefficient omitted*: $(s_{j,0}, s_{j,2}, \dots, s_{j,\ell})$.
2. $\mathcal{V}$ recovers the linear coefficient from the running claim $T_j = s_j(0) + s_j(1)$,
$$s_{j,1} = T_j - 2s_{j,0} - \textstyle\sum_{i\ge 2} s_{j,i}.$$
3. $\mathcal{V} \to \mathcal{P}$: sample $r_j \leftarrow \mathbb{F}$ and set $T_{j+1} := s_j(r_j)$.

Final check: $\mathcal{V}$ accepts iff $g(r_1, \dots, r_\mu) = T_{\mu+1}$.

Fig. 2: The compressed sum-check protocol [74]: each round message omits its linear coefficient, which the verifier recovers from the running claim $T_j = s_j(0) + s_j(1)$.

Batching. Suppose we hold t claims $T^{(i)} = \sum_{x\in\{0,1\}^{\mu_i}} g_i(x)$ with possibly different variable counts $\mu_1 \le \cdots \le \mu_t$. Assume that 2 is invertible in $\mathbb{F}$, as it is for every Akita field, and set $M := \mu_t$. For each instance i, choose an order-preserving injection $\pi_i : \{1, \dots, \mu_i\} \hookrightarrow \{1, \dots, M\}$ that places its variables among the common round positions, and extend g_i to M variables by ignoring the other coordinates:

$$\hat{g}_i(x_1, \dots, x_M) := g_i\big(x_{\pi_i(1)}, \dots, x_{\pi_i(\mu_i)}\big).$$

Every ignored coordinate doubles the Boolean sum, so the sum of $\hat{g}_i$ over $\{0,1\}^M$ is $2^{M-\mu_i}T^{(i)}$. After sampling $\gamma_1, \dots, \gamma_t \leftarrow \mathbb{F}$, the verifier batches the claims as

$$\sum_{i=1}^{t} \gamma_i 2^{M-\mu_i} T^{(i)} = \sum_{x\in\{0,1\}^M} \left(\sum_{i=1}^{t} \gamma_i \hat{g}_i(x) \right),$$

and the protocol runs one M-round sum-check on the right-hand side.

In round j, let $C_j^{(i)}$ be the running claim of instance i, with $C_1^{(i)} := 2^{M-\mu_i}T^{(i)}$. If j lies in the image of π_i, the instance contributes its ordinary degree-ℓ round polynomial. Otherwise $\hat{g}_i$ does not depend on the current variable, so its round polynomial is the constant $s_j^{(i)}(X) = C_j^{(i)}/2$: it satisfies $s_j^{(i)}(0) + s_j^{(i)}(1) = C_j^{(i)}$, and the next claim is $C_{j+1}^{(i)} = C_j^{(i)}/2$ for every challenge r_j. The prover sends the combined round polynomial $s_j = \sum_i \gamma_i s_j^{(i)}$, and the final check is the corresponding linear combination of $g_i\big(r_{\pi_i(1)}, \dots, r_{\pi_i(\mu_i)}\big)$. We use the conservative bound that this random linear combination adds at most $t/|\mathbb{F}|$ to the soundness error.

Akita uses the *suffix embedding* $\pi_i(k) = M - \mu_i + k$. The first $M - \mu_i$ rounds are then constant rounds for instance i, and its final evaluation uses the suffix $(r_{M-\mu_i+1}, \dots, r_M)$ of the shared challenge vector.

3.5 Equality-Factored Sum-Check

We use the equality-factored sum-check of [50,37] for instances of the form

$$T = \sum_{x\in\{0,1\}^\mu} \widetilde{\mathsf{eq}}(\tau, x)\, p(x),$$

where $\tau \in \mathbb{F}^\mu$ is fixed and the data polynomial p has individual degree at most $\ell - 1$.

The equality polynomial factors across coordinates, and we remove each equality factor from the running claim as its coordinate is processed. In round j, let q_j be the degree-$\le \ell - 1$ univariate obtained by binding the earlier coordinates and summing p against the equality factors of the later ones. Then

$$T_j = (1 - \tau_j)q_j(0) + \tau_j q_j(1) = q_{j,0} + \tau_j \sum_{i=1}^{\ell-1} q_{j,i}, \qquad T_1 := T, \quad T_{j+1} := q_j(r_j).$$

Since $q_{j,0}$ has coefficient one here, the prover omits the constant coefficient, rather than the linear one, and sends $(q_{j,1}, \ldots, q_{j,\ell-1})$. The verifier recovers

$$q_{j,0} = T_j - \tau_j \sum_{i=1}^{\ell-1} q_{j,i}$$

by multiplication and subtraction alone, for every $\tau_j \in \mathbb{F}$ including $\tau_j = 0$. Each round sends $\ell - 1$ coefficients, one fewer than the standard degree-ℓ format. The factorization also lets the prover store equality weights for the two halves of τ, about $2^{\mu/2}$ values each, instead of the full vector [50,37].

The coefficient omission requires the entire round message to carry one common, verifier-known equality factor. It therefore applies only when a sum-check runs alone or is batched with instances that share the same $\widetilde{\mathsf{eq}}(\tau, \cdot)$; otherwise only the prover-side storage saving remains. Section 6.1 states where Akita uses this format, and Table 4 collects the resulting message counts.

3.6 Quotient-Lift Ring Switching

Sum-check (Section 3.4) runs over a finite field, whereas Akita's root relation is defined over the cyclotomic ring $R_q = \mathbb{Z}_q[X]/(X^d+1)$. Ring switching, introduced by Huang, Mao, and Zhang [54] and used in Hachi [80], lifts an R_q-relation to $\mathbb{Z}_q[X]$ and evaluates the resulting identity at a random point $X = \alpha$ to obtain a finite-field claim.

The lift. Division by the monic polynomial $X^d + 1$ in $\mathbb{Z}_q[X]$ has a unique quotient and remainder. Write $\tilde{w} \in \mathbb{Z}_q^{<d}[X]$ for the canonical degree-$< d$ representative of $w \in R_q$, extended coordinatewise to vectors and matrices.

Lemma 3.4 (Ring-switch lift). *Let $\mathsf{M} \in R_q^{n\times m}$, $\mathbf{w} \in R_q^m$, and $\mathbf{h} \in R_q^n$. Then $\mathsf{M}\mathbf{w} = \mathbf{h}$ in R_q^n if and only if there exists $\mathbf{r} \in (\mathbb{Z}_q^{<d-1}[X])^n$ with*

$$\tilde{\mathsf{M}}\,\tilde{\mathbf{w}} = \tilde{\mathbf{h}} + (X^d + 1)\,\mathbf{r} \qquad \text{in } (\mathbb{Z}_q[X])^n.$$

The quotient $\mathbf{r}$ is unique and equals $(\tilde{\mathsf{M}}\tilde{\mathbf{w}} - \tilde{\mathbf{h}})/(X^d+1)$; each coordinate of $\tilde{\mathsf{M}}\tilde{\mathbf{w}} - \tilde{\mathbf{h}}$ has degree at most $2d-2$, so $\deg \mathbf{r} \le d-2$.

Proof. $\mathsf{M}\mathbf{w} = \mathbf{h}$ in R_q holds iff $\tilde{\mathsf{M}}\tilde{\mathbf{w}} - \tilde{\mathbf{h}} \equiv \mathbf{0} \pmod{X^d+1}$ coordinatewise, iff X^d+1 divides each coordinate in $\mathbb{Z}_q[X]$, iff such an $\mathbf{r}$ exists. Uniqueness and the degree bound are Euclidean division of a degree-$\le 2d-2$ element by the degree-d modulus X^d+1.

Soundness of evaluating at a random point. For $\alpha \in E$, the evaluation map $\varphi_\alpha : \mathbb{Z}_q[X] \to E$, $X \mapsto \alpha$, is a ring homomorphism, so applying it coordinatewise to Lemma 3.4 gives the identity over E

$$\mathsf{M}(\alpha)\,\mathbf{w}(\alpha) = \mathbf{h}(\alpha) + (\alpha^d + 1)\,\mathbf{r}(\alpha), \tag{24}$$

where $\mathbf{w}(\alpha) := \varphi_\alpha(\tilde{\mathbf{w}})$ and similarly for the other terms. Conversely, if $\mathsf{M}\mathbf{w} \neq \mathbf{h}$ then for *any* prover-supplied $\mathbf{r}$ with $\deg \mathbf{r} < d$ the vector $\mathbf{p}(X) := \tilde{\mathsf{M}}\tilde{\mathbf{w}} - \tilde{\mathbf{h}} - (X^d+1)\,\mathbf{r}$ is nonzero in some coordinate and has degree at most $2d-1$, so by Schwartz–Zippel

$$\Pr_{\alpha \leftarrow E}\big[\mathbf{p}(\alpha) = \mathbf{0}\big] \le \frac{2d-1}{|E|}.$$

Choosing k so that q^k is exponential in λ makes this negligible.

Evaluation as a weighted sum over coefficients. The prover commits once to the base-field coefficient vector of the witness and never commits to $\mathbf{w}(\alpha)$. Pad m to a power of two, $2^{\mu_x} = m$. Index a ring coordinate by $x \in \{0,1\}^{\mu_x}$ and a coefficient $0 \le \ell < d$ by $y \in \{0,1\}^{\log d}$, and let $\widetilde{\mathbf{w}} : \mathbb{F}_{q^k}^{\mu_x} \times \mathbb{F}_{q^k}^{\log d} \to \mathbb{F}_{q^k}$ be the multilinear extension of the function $\mathbf{w}(x,y)$ that returns coefficient $\ell(y)$ of ring coordinate x. The multilinear extension of $\ell \mapsto \alpha^\ell$ is the product

$$\widetilde{\alpha}(y) = \prod_{j=0}^{\log d - 1} \big(1 + y_j(\alpha^{2^j} - 1)\big) \in \mathbb{F}_{q^k}, \tag{25}$$

since the right-hand side is multilinear and equals $\alpha^{\ell(y)}$ on the Boolean cube. Hence, for every coordinate x,

$$\mathbf{w}_x(\alpha) = \sum_{\ell<d} \mathbf{w}(x, y(\ell))\, \alpha^{\ell} = \sum_{y\in\{0,1\}^{\log d}} \widetilde{\mathbf{w}}(x, y)\, \widetilde{\alpha}(y). \tag{26}$$

Evaluation at $X = \alpha$ is thus a sum of $\widetilde{\mathbf{w}}$ over its coefficient variables with the public weight $\widetilde{\alpha}$, which a sum-check can verify against the committed vector.

3.7 Tensor Reduction to the Extension Field

Diamond and Posen [39, Theorem 3.5] reduce an evaluation claim on a base-field multilinear polynomial at an extension-field point to a single evaluation claim on a packed extension-field polynomial with κ fewer variables, where 2^κ is the extension degree. We call this protocol the *tensor reduction*, to distinguish it from the ring switching of Section 3.6. Akita applies it when a fold that uses the evaluation-trace opening method receives an ordinary base-field opening; its transcript and soundness error are charged to that receiving fold.

Packed polynomial. Fix $\kappa \geq 1$ and a deterministic $\mathbb{F}_q$-basis $(\beta_y)_{y\in\{0,1\}^\kappa}$ of $E = \mathbb{F}_{q^{2^\kappa}}$. Split the ℓ variables of a base-field multilinear polynomial $f \in \mathbb{F}_q^{\leq 1}[X_{\mathsf{head}}, X_{\mathsf{tail}}]$ into κ head variables and $\ell - \kappa$ tail variables, and pack the head variables into the basis:

$$\bar{f}(X_{\mathsf{tail}}) := \sum_{y\in\{0,1\}^\kappa} f(y, X_{\mathsf{tail}})\, \beta_y \in E^{\leq 1}[X_{\mathsf{tail}}].$$

The input is a claim $f(r_{\mathsf{head}}, r_{\mathsf{tail}}) = v$ with $r_{\mathsf{head}} \in E^\kappa$, $r_{\mathsf{tail}} \in E^{\ell-\kappa}$, and $v \in E$. The output is one claim on $\bar{f}$ at a point $\rho \in E^{\ell-\kappa}$ produced by the reduction.

The reduction. The prover sends the *column partials* $S_y := f(y, r_{\mathsf{tail}}) \in E$ for $y \in \{0,1\}^\kappa$, and the verifier checks $v = \sum_y \widetilde{\mathsf{eq}}(y, r_{\mathsf{head}}) S_y$. Checking only $\sum_y \beta_y S_y$ would be insecure: the basis combination binds $\mathbb{F}_q$-coordinates, not arbitrary E-coefficients. To bind the S_y to $\bar{f}$, write $S_y = \sum_u S_{u,y}\beta_u$ with $S_{u,y} \in \mathbb{F}_q$ and form the *row partials* $row_u := \sum_y S_{u,y}\beta_y$ for $u \in \{0,1\}^\kappa$. Let $A_u(w) \in \mathbb{F}_q$ be the coordinates of the tail equality factor, $\widetilde{\mathsf{eq}}(r_{\mathsf{tail}}, w) = \sum_u A_u(w)\beta_u$, and let A_u also denote the multilinear extension of $w \mapsto A_u(w)$. The honest row partials satisfy

$$row_u = \sum_{w\in\{0,1\}^{\ell-\kappa}} A_u(w)\, \bar{f}(w), \qquad u \in \{0,1\}^\kappa. \tag{$\star$}$$

The verifier batches these claims with $\eta \in E^\kappa$, $\eta_u := \widetilde{\mathsf{eq}}(u, \eta)$, setting $c_\eta := \sum_u \eta_u row_u$ and $A_\eta := \sum_u \eta_u A_u$, so $c_\eta = \sum_w A_\eta(w)\bar{f}(w)$. A degree-2 sum-check (Figure 2) over the $\ell - \kappa$ tail variables then yields a point ρ and a final value $final = A_\eta(\rho)\bar{f}(\rho)$. The verifier computes $A_\eta(\rho)$ itself: in the tensor algebra $E \otimes_{\mathbb{F}_q} E$, with embeddings $\phi_0(\alpha) := \alpha \otimes 1$ and $\phi_1(\alpha) := 1 \otimes \alpha$, write $e := \widetilde{\mathsf{eq}}(\phi_0(r_{\mathsf{tail}}), \phi_1(\rho)) = \sum_u \beta_u \otimes e_u$; then $A_\eta(\rho) = \sum_u \eta_u e_u$, computed in $O(\ell - \kappa)$ tensor-algebra operations [39]. The verifier rejects if $A_\eta(\rho) = 0$, since the final equation would then hold for every value of $\bar{f}(\rho)$. Otherwise it checks $final = A_\eta(\rho)\bar{f}(\rho)$ and outputs the claim $\bar{f}(\rho)$. An honest prover is rejected with probability at most $\ell/|E|$, and a false input claim passes with probability at most $(\kappa + 2(\ell - \kappa))/|E|$; Theorem 5.1 proves these bounds for the batched form that Akita uses.

3.8 Polynomial Commitments and Coordinate-Wise Special Soundness

We fix the polynomial-commitment syntax and security notions, and then recall the coordinate-wise special-soundness framework, which Section 10.3 extends.

Definition 3.5 (Multilinear PCS). *Let K be a finite field and let E/K be a finite extension. A (K, E)-*multilinear polynomial commitment scheme *consists of four algorithms:*

(i) $\mathsf{Setup}(1^\lambda, \mu_{\max}) \to \mathsf{pp}$;

(ii) $\mathsf{Commit}(\mathsf{pp}, \mathsf{shape}, f) \to (\mathsf{cm}, \mathsf{st})$, *where the admissible descriptor* shape *fixes an arity* $\mu \le \mu_{\max}$, $f : \{0,1\}^\mu \to K$ *records the Boolean evaluation vector in the fixed index order, and* $\mathsf{cm} := (\mathsf{shape}, C)$ *is the public commitment;*
(iii) $\mathsf{Eval.P}(\mathsf{pp}, \mathsf{cm}, \mathsf{st}, \mathbf{r}, v) \to \pi$*;*
(iv) $\mathsf{Eval.V}(\mathsf{pp}, \mathsf{cm}, \mathbf{r}, v, \pi) \to \{0,1\}$, *where* $\mathbf{r} \in E^\mu$ *and* $v \in E$.
It also specifies an efficiently decidable decommitment relation

$$\mathsf{Rel}_{\mathsf{com}}(\mathsf{pp}, \mathsf{cm}, f, \mathsf{st}) \in \{0,1\}.$$

For every output $(\mathsf{cm}, \mathsf{st}) \leftarrow \mathsf{Commit}(\mathsf{pp}, \mathsf{shape}, f)$*, this relation must accept.*

Knowledge soundness is stated through $\mathsf{Rel}_{\mathsf{com}}$, an algebraic decommitment relation, rather than by rerunning a possibly randomized Commit (Remark 10.41). The evaluation relation is

$$\mathsf{Rel}_{\mathsf{eval}}(\mathsf{pp}, \mathsf{cm}, \mathbf{r}, v; f, \mathsf{st}) = 1 \iff \mathsf{Rel}_{\mathsf{com}}(\mathsf{pp}, \mathsf{cm}, f, \mathsf{st}) = 1 \ \wedge \widetilde{f}(\mathbf{r}) = v. \tag{27}$$

The shape carried by cm determines the type of every argument in this relation, so the payload C is never interpreted without its shape. We require completeness and knowledge soundness.

Definition 3.6 (PCS completeness). *The scheme has completeness error* $\varepsilon_{\mathsf{comp}}$ *if, for every admissible* shape*, every Boolean evaluation vector* $f : \{0,1\}^\mu \to K$ *of that shape, and every* $\mathbf{r} \in E^\mu$*,*

$$\Pr\left[\begin{array}{c} \mathsf{Eval.V}(\mathsf{pp}, \mathsf{cm}, \mathbf{r}, \widetilde{f}(\mathbf{r}), \pi) = 1 \\ \Big| \\ \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, \mu_{\max}) \\ (\mathsf{cm}, \mathsf{st}) \leftarrow \mathsf{Commit}(\mathsf{pp}, \mathsf{shape}, f) \\ \pi \leftarrow \mathsf{Eval.P}(\mathsf{pp}, \mathsf{cm}, \mathsf{st}, \mathbf{r}, \widetilde{f}(\mathbf{r})) \end{array}\right] \ge 1 - \varepsilon_{\mathsf{comp}}(\lambda).$$

Perfect completeness is the special case $\varepsilon_{\mathsf{comp}}(\lambda) = 0$.

In the knowledge-soundness experiment, $\mathsf{Run}[\mathcal{P}^*(\mathsf{pp})]$ records the displayed statement and proof together with a resettable execution handle hdl. The handle fixes the prover's auxiliary input, initial state, and random tape, and in the random-oracle model also the oracle transcript of that execution. Thus $\mathcal{E}^{\mathsf{hdl}}$ replays or resumes this execution rather than an independently sampled copy of $\mathcal{P}^*$, and answers replayed oracle queries consistently except at the challenge queries that its rewinding resamples.

Definition 3.7 (PCS knowledge soundness). *The scheme has knowledge error* κ_{ks} *if, for every stateful PPT prover* $\mathcal{P}^*$*, there exists an expected polynomial-time extractor* $\mathcal{E}$ *with resettable black-box access to* $\mathcal{P}^*$ *such that*

$$\Pr\left[\begin{array}{c} \mathsf{Eval.V}(\mathsf{pp}, \mathsf{cm}, \mathbf{r}, v, \pi) = 1 \\ \wedge \ \big((f^*, \mathsf{st}^*) = \bot \ \vee \ \mathsf{Rel}_{\mathsf{eval}}(\mathsf{pp}, \mathsf{cm}, \mathbf{r}, v; f^*, \mathsf{st}^*) = 0\big) \\ \Big| \\ \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, \mu_{\max}) \\ (\mathsf{cm}, \mathbf{r}, v, \pi; \mathsf{hdl}) \leftarrow \mathsf{Run}[\mathcal{P}^*(\mathsf{pp})] \\ (f^*, \mathsf{st}^*) \leftarrow \mathcal{E}^{\mathsf{hdl}}(\mathsf{pp}, \mathsf{cm}, \mathbf{r}, v, \pi) \end{array}\right] \le \kappa_{\mathsf{ks}}(\lambda).$$

Coordinate-wise special soundness (CWSS). We analyze the folding step at each recursive level in the CWSS framework of Fenzi, Moghaddas, and Nguyen [43], together with its Fiat–Shamir analysis in the random-oracle model.

Definition 3.8 (Special-sound structure [80, Definition 3]). *Let* $\mathcal{C}_1, \dots, \mathcal{C}_\mu$ *be challenge spaces. For vectors* $\ell = (\ell_1, \dots, \ell_\mu)$ *and* $k = (k_1, \dots, k_\mu)$ *with* $k_i \ge 2$*, write* $N_{\mathsf{SS}} := \prod_{i=1}^{\mu} (\ell_i(k_i - 1) + 1)$*. The set* $\mathsf{SS}(\mathcal{C}, \ell, k)$ *consists of all families of* N_{SS} *challenge tuples* $\{\mathbf{c}^{(j)}\}_{j \in [N_{\mathsf{SS}}]} \subseteq \mathcal{C}_1^{\ell_1} \times \cdots \times \mathcal{C}_\mu^{\ell_\mu}$ *arranged in a* μ*-level tree with the root at depth zero. For* $i \in \{1, \dots, \mu\}$*, each node at depth* $i-1$ *has* $\ell_i(k_i - 1) + 1$ *children. Its set of sibling challenge vectors in* $\mathcal{C}_i^{\ell_i}$ *contains, for each coordinate position* $p \in [\ell_i]$*, at least* k_i *vectors that are pairwise distinct in position* p *and identical in all other positions.*

Definition 3.9 (CWSS protocol [80, Definition 3]). *A public-coin interactive protocol with μ challenge rounds, where round i draws a challenge from $\mathcal{C}_i^{\ell_i}$, is $(\ell_1, \ldots, \ell_\mu)$*-coordinate-wise *$(k_1, \ldots, k_\mu)$*-special sound *if there exists a deterministic polynomial-time extractor that, given any prefix-consistent tree of N_{SS} accepting transcripts (the prover messages agree at every common prefix) whose challenges form a member of $\mathsf{SS}(\mathcal{C}, \ell, k)$, outputs a valid witness.*

Theorem 3.10 (CWSS-to-knowledge and classical Fiat–Shamir [43, Lemmas 2.31 and 2.32]). *Let Π be a public-coin $(2\mu+1)$-round interactive proof system for a relation R, and suppose that in each challenge round the verifier samples from $\mathcal{X}^\ell$. If Π is ℓ-coordinate-wise k-special sound and $(1+\ell(k-1))^\mu = \mathsf{poly}(\lambda)$, then Π is knowledge sound with knowledge error*

$$\frac{\mu\,\ell\,(k-1)}{|\mathcal{X}|},$$

and the extractor runs the malicious prover $(\ell(k-1)+1)^\mu$ times in expectation. Moreover, the Fiat–Shamir transform of Π is knowledge sound against a classical random-oracle adversary making Q queries, with knowledge error

$$(Q+1)\cdot\frac{\mu\,\ell\,(k-1)}{|\mathcal{X}|}.$$

This statement is not a QROM theorem.

4 Setup and Commitments

An application can commit before it knows the opening points or the other commitments that will join its proof. We therefore keep commitment formation separate from the fold: each commitment fixes its own representation and binding parameters, while the fold of Section 5 chooses compatible opening data for the complete batch. This section fixes the committed objects, the public schedule connecting successive folds, the shared setup, and the commitment maps. Tables 8 and 9 in Appendix A collect the notation of Section 4 through Section 9.

4.1 Setting

Object. Let $K := \mathbb{F}_q$ and $E := \mathbb{F}_{q^k}$. The protocol commits to a Boolean evaluation vector $f : \{0,1\}^\mu \to K$, which determines the μ-variate multilinear polynomial $\widetilde{f}$. It later proves one evaluation claim

$$\widetilde{f}(\mathbf{r}) = v, \qquad \mathbf{r} \in E^\mu, \qquad v \in E,$$

matching the PCS interface of Definition 3.5. Thus the commitment uses base-field coefficients even when the opening point and claimed value use the extension field. This is the $P = 1$ interface. More generally, one commitment can bind an ordered tuple $(f_p)_{p\in[P]}$ of such vectors: its source layout fixes P, their order, and their common commitment grid before any opening request. The public commitment pairs its shape with its payload: $\mathsf{cm} = (\mathsf{shape}_{\mathsf{com}}, \mathbf{u}_{\mathsf{pub}})$; the payload is never interpreted without its frozen shape.

Each fold uses the opening method that (127) prescribes for its level. *Direct coefficient packing* retains one coefficient axis in a challenge subring and opens the mixed-field claim directly (Section 5.3). The *evaluation-trace* method first applies, for $k > 1$, the tensor reduction of Section 3.7 and then the evaluation-trace relation. Both methods produce the same ordinary recursive opening. Section 9.3 assigns methods to nonterminal folds; the terminal uses evaluation trace, whose tensor reduction is the identity when $k = 1$.

Coefficients. Fix the ring-subfield K-basis $\boldsymbol{\beta} = (\beta_0, \ldots, \beta_{k-1})$ of E from Section 3.1. Extension-field values arise only after the opening contraction. They are expanded as $u = \sum_{t\in[k]} \beta_t u_t$ with $u_t \in K$ whenever they enter a commitment, range check, transcript encoding, or witness layout. Throughout this section, *coefficient* means one such base-field coordinate. The digit and one-hot restrictions below apply to these coordinates, not to the monomial coefficients of $\widetilde{f}$; the packing map can change their bounds when it forms a packed ring element.

Witness shapes. The vector f carries a *shape*, fixed when the commitment is formed. Together with the packing map, the shape determines the source encoding and its packed-coefficient bound (Section 4.5). We support the three shapes below; further shapes can be added through the same gadget parameter.

1. **Full-field.** Arbitrary coefficients in $\mathbb{F}_q$. The commitment digit-decomposes every coefficient (gadget depth $\delta_{\mathsf{com}} = \lceil \log_b q \rceil$).
2. **Small balanced digits.** Coefficients already lie in the balanced alphabet $\{-b/2, \ldots, b/2 - 1\}$. This is the shape of every recursive witness (Section 5.5). We use a depth-1 identity source map, with the packed-coefficient bound obtained after the selected packing.
3. **One-hot.** The evaluation vector is a concatenation of blocks, each containing exactly one coefficient equal to 1 and zeros elsewhere (a 1-of-2^κ guarantee). We use a depth-1 identity source map and certify its packed-coefficient bound. The root schedule uses this occupancy in the fixed-weight moment-generating-function bound of Theorem K.5; treating the shape as a generic dense source with coefficient bound one is valid but looser.

Remark 4.1 (Shape guarantees are conditional). The digit range check (Section 6.1) validates the small-digit shape inside the protocol. The surrounding proof system validates the one-hot shape (in Jolt, by booleanity and Hamming-weight sum-checks before the batched opening). Both validations are conditional on the PCS claim. Akita uses the one-hot guarantee only to sharpen the honest-response bound: its extractor returns a polynomial with the algebraic decommitment of Definition 10.21, whether or not that polynomial is one-hot.

Commitment shape. We freeze the source layout together with the matrix parameters so that a later opening cannot change what the commitment represents. The commitment shape is

$$\mathsf{shape}_{\mathsf{com}} := \big(\mathsf{Src};\ M, b, b_1; (d_A, n_A); (d_B, n_B, S_B); \mathcal{F}\big). \tag{28}$$

The source layout Src fixes the number P and order of the committed evaluation vectors, their natural lengths, their common commitment grid and coordinate order, the honest padding embeddings, the source coefficient constraints, and the decoding map used in Definition 10.21. Recording an honest padding embedding does not assert that every algebraic decommitment lies in its image. Together with d_A, Src determines the ring-packed length N of each vector and hence the block count $B = \lceil N/M \rceil$. All P vectors use one declared coefficient envelope and one source digit depth δ_{com}, so their blocks have the same $\mathbf{A}$ input width; vectors requiring different source digit depths use separate commitment handles. Opening points, claimed values, and the selection of point coordinates belong to the later opening statement.

Here M is the block length; b and b_1 are the commitment and opening bases; (d_A, n_A) and (d_B, n_B) give the ring dimensions and module ranks of the inner and outer maps; and S_B is the frozen dyadic slice count from which the canonical ranges $\Pi_B(B, S_B)$ of (43) are derived. The compression profile $\mathcal{F}$ is empty when the outer image is transmitted directly. Otherwise it contains exactly the two triples $(\{-1, 0\}, d_{F,i}, n_{F,i})_{i \in \{1,2\}}$ for the two binary compression maps, whose input widths follow from the outer image and from the first compressed image.

Fields and matrix notation. We use $k \in \{1, 2, 4\}$ and write $R_{q,d} = \mathbb{Z}_q[X]/(X^d + 1)$ to keep each map's ring dimension visible. A matrix over R_{q,d_I} has n_I rows, determined by the Module-SIS security requirement, and m_I columns, determined by the length of its input. Sections 4.3 and 4.5 give the five matrix types and construct their inputs.

4.2 Schedule Topology and Fold Configuration

The protocol is a sequence of t *folds* followed by a *terminal level*. A public description has three layers: the opening groups entering a fold, the four fields that configure the fold, and the edge that connects its output to the next level. Root batching changes the groups entering the first fold, whereas setup offloading changes the claims passed from a fold to its successor.

Opening groups and source topology. An opening instance consists of a reference to an existing commitment, one evaluation point, and the values claimed at that point. We call the referenced pair of frozen shape and payload a *commitment handle*. Referencing a handle twice does not rerun the commitment algorithm;

opening the same commitment at a second point creates a second instance. A source group is one such instance occupying one position in the ordered input to a fold. An *opening shape* records the public form of this ordered list, but not the commitments, points, claimed values, or witness entries.

The root source is an ordered, nonempty list of application-supplied groups,

$$\mathcal{O}_0 = (\mathcal{A}_{0,0}, \dots, \mathcal{A}_{0,G_0-1}). \tag{29}$$

Taking $G_0 = 1$ with one claim and no padding to a larger grid gives the ordinary single-opening PCS interface. Application-supplied multi-group batching is permitted only at the root. Afterwards, the preceding edge determines the source: fold $h > 0$ receives the recursive-witness group $\mathcal{W}_h$, and also the setup-prefix group $\mathcal{S}_{h-1}$ if fold $h-1$ offloaded its setup evaluation:

$$\mathcal{O}_h = \begin{cases} (\mathcal{W}_h), & \mathsf{edge}_{h-1} = \mathsf{Direct}, \\ (\mathcal{S}_{h-1}, \mathcal{W}_h), & \mathsf{edge}_{h-1} = \mathsf{OffloadedSetup}. \end{cases} \tag{30}$$

Here $\mathcal{S}_{h-1} = (C_{S,h-1}, \rho_{S,h-1}, s_{\rho_{S,h-1}})$ is the opening claim on the public setup prefix produced by the setup-product sum-check of fold $h-1$; its payload $C_{S,h-1}$ belongs to the preprocessed slot fixed by the schedule. The next fold checks this group, and it does not become a permanent component of the recursive state. Thus every non-root fold receives at most two groups.

Four configuration fields. A *fold configuration* records four fields:

$$\mathsf{cfg}_h := \big(\mathsf{Open}_h, \mathsf{RingCheck}_h, \mathsf{Comp}_h, \mathsf{Norm}_h\big). \tag{31}$$

These fields select the opening method, ring-reduction route, compression profile, and response-norm certificate. They summarize the exact records of Section 5.1 and are not serialized separately. The policy (127) determines Open_h, and Definition 9.2 constrains the remaining fields. With the source and target shapes fixed, the configuration determines the next witness (Section 5.5), its norm checks (Section 6), and the relation rows (Section 7.2).

Edges and schedules. Every committed fold records an outgoing edge tag

$$\mathsf{edge}_h \in \{\mathsf{Direct}, \mathsf{OffloadedSetup}, \mathsf{TerminalInnerState}\}. \tag{32}$$

A direct edge produces only $\mathcal{W}_{h+1}$. An offloaded edge produces $(\mathcal{S}_h, \mathcal{W}_{h+1})$ and requires the successor to check both groups. The final committed fold instead produces the canonical inner state checked by the terminal; it cannot leave a setup group pending.

A *level description* consists of a source opening shape, a fold configuration, the remaining public parameters of that fold, an outgoing edge tag, and the resulting target shape. A *schedule* is a finite composable sequence

$$\begin{aligned} \mathcal{O}_0 \xrightarrow{\mathsf{Fold}_0[\mathsf{cfg}_0;\mathsf{edge}_0]} \mathcal{O}_1 \xrightarrow{\mathsf{Fold}_1[\mathsf{cfg}_1;\mathsf{edge}_1]} \cdots \\ \xrightarrow{\mathsf{Fold}_{t-1}[\mathsf{cfg}_{t-1};\mathsf{TerminalInnerState}]} \mathcal{T}_t \xrightarrow{\mathsf{Terminal}} \mathsf{accept}. \end{aligned} \tag{33}$$

For every nonfinal edge, the target shape is exactly the source shape in (30) selected by that edge. This equality is part of the syntax; the conditions that make a schedule admissible are stated in Section 9.3. Section 5 constructs one fold for all these source lists, and Section 9 supplies the setup-product stage of an offloaded edge. Distributed proving refines how a level is executed and does not change this topology (Section 11).

Fiat–Shamir profile. Every schedule records a target λ_{FS} and a maximum number $Q_{\max}$ of classical queries to the Fiat–Shamir oracle $\mathcal{H}_{\mathsf{FS}}$, counting every nonce probe; the security claim covers only adversaries within this count. Each source group records its challenge rule (Section 5.1), including a certified lower bound on $|\mathcal{C}^{\mathsf{acc}}|$ and a pairwise-difference certificate. Every response-producing stage, including the terminal, records a nonce set of cardinality $N_{\mathsf{try},h}$ and a deterministic joint retry rule. The schedule digest binds these values and all opening shapes and level descriptions, and is absorbed before the first challenge (Figure 7). Within a fold, each challenge coordinate is derived from a separately indexed query under a fixed round root and candidate nonce, which makes the coordinate-wise forks of Section 10 explicit. An algebraic proof-of-work profile also binds its ordered predicate sites, targets, and nonce widths (Section 10.1).

4.3 Setup

The commitment and opening algorithms use several public matrices, whose dimensions can differ across the recursion. We read all of them from one public vector $\mathbf{S} = (S_0, \ldots, S_{N_{\mathsf{setup}}-1}) \in \mathbb{Z}_q^{N_{\mathsf{setup}}}$, expanded from a public seed. Each matrix uses a prefix of this vector, so the setup need only accommodate the largest matrix.

Matrix inventory. We distinguish five matrix types, $\mathbf{A}, \mathbf{B}, \mathbf{D}, \mathbf{F}, \mathbf{H}$. A matrix instance fixes one type, one fold level, one scheduled shape, and, for $\mathbf{F}/\mathbf{H}$, one compression-chain position. Each group reuses its commitment matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{F}$, and the fold adds one shared opening path $\mathbf{D}/\mathbf{H}$. Suppressing the group index and the level superscript (h) (as in $\mathbf{A}^{(h)}$):

- $\mathbf{A} \in R_{q,d_A}^{n_A \times m_A}$: inner commitment;
- $\mathbf{B} \in R_{q,d_B}^{n_B \times m_B}$: outer commitment, applied to each slice after padding to width m_B;
- $\mathbf{D} \in R_{q,d_D}^{n_D \times m_D}$: one unsliced partial-evaluation commitment, applied to the complete ordered partial digit vector;
- binary compression matrices $\mathbf{F}_i \in R_{q,d_{F,i}}^{n_{F,i} \times m_{F,i}}$ and $\mathbf{H}_i \in R_{q,d_{H,i}}^{n_{H,i} \times m_{H,i}}$, for $1 \le i \le L_F = L_H \in \{0, 2\}$, all over the alphabet $\{-1, 0\}$ (Section 4.5).

Length and matrix entries. Fix the maximum supported variable count $\mu_{\max}$, and let

$$N_{\mathsf{view}} := \max\left\{ n_I m_I d_I : \begin{array}{l} I \text{ is a matrix instance in an admitted schedule} \\ \text{with } \mu \le \mu_{\max} \end{array} \right\}.$$

The schedule family is finite and of bounded depth, so N_{view} is computed once, offline, rather than defined recursively at proof time. The provisioned stream length is its least power-of-two envelope

$$N_{\mathsf{setup}} \;:=\; 2^{\lceil \log_2 N_{\mathsf{view}} \rceil}. \tag{34}$$

A matrix $\mathbf{M} \in R_{q,d}^{n \times m}$ uses the first nmd entries of $\mathbf{S}$, in row-major order, with the d coefficients of each ring element consecutive:

$$\mathbf{M}[i, j] \;:=\; \sum_{\ell=0}^{d-1} S_{(im+j)\,d\,+\,\ell}\, X^{\ell} \;\in\; R_{q,d}, \qquad 0 \le i < n,\ 0 \le j < m. \tag{35}$$

We call this prefix interpretation the *matrix view* of $\mathbf{M}$. Matrices of different types or levels therefore reuse entries of $\mathbf{S}$ under different values of (n, m, d). Columns follow the order of the committed vector: the $\mathbf{B}$ view uses the polynomial-major slice input of (44), and the shared $\mathbf{D}$ view concatenates the opening-digit intervals in group order, with the addresses fixed in Appendix L.2.

Seed expansion. The vector $\mathbf{S}$ is expanded from a public 256-bit seed by a domain-separated extendable-output function as one coefficient-indexed flat stream:

$$S_i := \mathsf{XOF}_q(\mathsf{seed}; \mathsf{setup}, i), \qquad i = 0, 1, 2, \ldots \tag{36}$$

The public setup profile fixes the concrete function, its domain-separation labels, and its map from sampled bytes to $\mathbb{Z}_q$. The derivation is prefix-consistent: a length-N expansion is a prefix of any longer expansion from the same seed, as the overlapping prefix views require. Each coefficient is within statistical distance $q\,2^{-w}$ of uniform, where the draw width w is at least twice the modulus bit-length (the largest moduli sample exactly, by rejection). The aggregate bias $N_{\mathsf{setup}}\, q\, 2^{-w}$ enters the advantage formula of Section 10, where the expansion is modeled as a random oracle. The setup profile records the actual sampling map and width and checks this bound for the largest admitted prefix; Appendix L.1 gives a sufficient width for a target aggregate error. We write $\mathcal{H}_{\mathsf{setup}}$ and $\mathcal{H}_{\mathsf{FS}}$ for the restrictions of the random oracle to the setup-expansion and Fiat–Shamir namespaces, which domain separation makes independent. Queries that recompute public setup coefficients use $\mathcal{H}_{\mathsf{setup}}$ and do not count toward $Q_{\max}$.

The public parameters are $\mathsf{pp} := (\mathsf{seed}, \mathsf{sch}, \mathsf{Reg})$: the seed, the schedule sch, and the registry Reg of offloaded setup commitments (Section 9), which is empty when no edge is offloaded.

4.4 Per-Matrix Ring Dimensions and Challenge Families

The matrices used by Akita need not share a ring dimension: by Lemma 7.1, the relations can use different quotient rings over the same flat coefficient vector. This subsection gives the compatibility conditions imposed by the opening method and its challenges; Section 12.2 explains how the planner compares compatible dimensions.

Challenge subring. At a fold using direct coefficient packing (Section 5.3), let d_{car} and h_{car} be powers of two with $d_A = kh_{\mathsf{car}}d_{\mathsf{car}}$, where d_A is the dimension of the inner commitment matrix. We use the three rings

$$S := \mathbb{F}_q[Y]/(Y^{d_{\mathsf{car}}}+1), \qquad S_E := E[Y]/(Y^{d_{\mathsf{car}}}+1), \qquad R := \mathbb{F}_q[X]/(X^{d_A}+1), \tag{37}$$

the *challenge subring*, the *extension-field carrier ring*, and the *A ring* over which $\mathbf{A}$ is defined. Since $(X^{kh_{\mathsf{car}}})^{d_{\mathsf{car}}} = X^{d_A} = -1$ in R, there is a canonical embedding

$$\iota : S \longrightarrow R, \qquad Y \longmapsto X^{kh_{\mathsf{car}}}, \tag{38}$$

under which the A ring is a free S-module of rank kh_{car}:

$$R = \bigoplus_{a=0}^{kh_{\mathsf{car}}-1} X^a \iota(S). \tag{39}$$

Lemma 4.2 (Challenge-subring embedding). *For every $c \in S$, the embedding (38) has the following properties.*

1. *If c is a unit in S, then $\iota(c)$ is a unit in R.*
2. *The coefficient norms are preserved:*

$$\|\iota(c)\|_p = \|c\|_p \qquad \text{for } p \in \{1, 2, \infty\}.$$

3. *Multiplication has the same Euclidean operator norm in the challenge subring and in the A ring:*

$$\|\iota(c)\|_{\mathsf{mul},2,R} = \|c\|_{\mathsf{mul},2,S}.$$

The same unit statement holds after extending scalars from S to S_E.

Proof. The embedding is unital, so it maps an inverse of c to an inverse of $\iota(c)$. It inserts the coefficients of c at positions divisible by kh_{car} and inserts zeros elsewhere, which proves the coefficient-norm identities. Under (39), multiplication by $\iota(c)$ is the direct sum of kh_{car} copies of negacyclic multiplication by c in S. A direct sum of identical linear maps has the same spectral norm as one copy. Scalar extension preserves the inverse identity.

At such a *packing fold*, the protocol samples each challenge from an audited family in the challenge subring of dimension d_{car} and embeds it into the A ring. The family must meet the schedule's Fiat–Shamir entropy target, and every nonzero pairwise difference must be a unit in S (Section 3.2). The family at d_{car} determines the challenge norm, the folded-response envelope, and the $\mathbf{A}$ collision radius. The A-ring dimension, subject to

$$kd_{\mathsf{car}} \mid d_A, \qquad h_{\mathsf{car}} = \frac{d_A}{kd_{\mathsf{car}}},$$

determines the embedding stride kh_{car}, the $\mathbf{A}$ module rank, its output quantisation, and its ring-arithmetic cost. An evaluation-trace fold instead samples its challenge in the full ring R_{q,d_A} and has no challenge-subring dimension. For either opening method, each claim–block pair receives an independent draw from the challenge family fixed for its level.

The commitment rings d_B, d_D. The outer and partial-evaluation consistency rows carry no challenge, and their collision bounds are the certified digit alphabet diameters (Remark 10.24). We therefore choose d_B and d_D by their required ranks and padded sizes, and either may lie below d_A. Both $\hat{\mathbf{t}}$ and $\hat{\mathbf{e}}$ remain flat K-coefficient vectors read under schedule-fixed ring views, so their digit bounds and commitment binding do not depend on those views (Remark 7.2). One packing partial has kd_{car} base-field coordinates and one evaluation-trace partial has d_A, so the ring views require

$$d_B \mid d_A, \qquad d_D \mid \begin{cases} kd_{\mathsf{car}} & \text{with direct coefficient packing,} \\ d_A & \text{with evaluation trace,} \end{cases}$$

and no condition $d_D \mid d_B$. Each partial occupies the corresponding number of **D** subcolumns; multiplying by digit depth and claim–block multiplicity gives the input width that sizes the **D** rank and the **H** compression widths.

Ring switching uses one ladder of powers α^{2^j} (Lemma 7.1), and every matrix uses the prefix matching its own native dimension. With coefficient packing, the embedded challenge evaluation for the A rows uses the stride prescribed by (114); evaluation trace uses the ordinary A-ring challenge evaluation. The dimensions may vary by level and, where the grouped interface permits, by group, and each induced commitment instance must meet its security conditions. The schedule fixes these choices, the opening method, challenge family, coordinate order, and terminal path before the first fold challenge.

Challenge norms. For a packing challenge c and an A-ring response u, Lemma 4.2 gives

$$\|\iota(c) * u\|_\infty \le \|c\|_1 \, \|u\|_\infty.$$

Thus the ℓ_1 norm ω of the challenge-subring family controls the deterministic response envelope and the coefficient-ℓ_∞ collision bound. Lemma 4.2 also supplies the multiplication-operator bound used by a Euclidean security analysis; whether the schedule enables that response certificate is a separate admissibility decision. The response depth and **A** collision bound are derived for the opening method prescribed at each level (Section 12.2).

4.5 Commitment

Akita uses a two-tier Ajtai commitment, instantiated separately at each scheduled ring dimension and extended with optional compression chains. A *payload* is the message actually sent in the interactive protocol (equivalently, included in the proof in the Fiat–Shamir form): either an Ajtai image or its compression through a further Ajtai chain. Figure 3 shows the pipeline.

Blocks. We describe one vector in the source layout and suppress its polynomial index. For an ordered tuple $(f_p)_{p\in[P]}$, the block construction and the inner commitment apply separately to every p, giving $\mathbf{f}_{p,i}$, $\mathbf{s}_{p,i}$, and $\mathbf{t}_{p,i}$; the sliced outer map then binds all inner images in the frozen polynomial order. Let f be the declared Boolean evaluation vector before padding, and write $\mathbf{f} \in R_{q,d_A}^N$ for its ring-packed representation, constructed below. The commitment shape fixes $M := 2^{r_{\mathsf{pos}}}$ ring elements per block, so $\mathbf{f}$ occupies

$$B := \left\lceil \frac{N}{M} \right\rceil \tag{40}$$

blocks. The honest commitment algorithm extends $\mathbf{f}$ by zero, so block $i \in [B]$ is

$$\mathbf{f}_i[x] := \begin{cases} \mathbf{f}[iM + x] & \text{if } iM + x < N, \\ 0 & \text{otherwise,} \end{cases} \qquad x \in [M]. \tag{41}$$

The block number is represented by $r_{\mathsf{blk}} := \lceil \log_2 B \rceil$ variables in a multilinear extension, so block numbers $B, \dots, 2^{r_{\mathsf{blk}}} - 1$ evaluate to zero without appearing in the committed vector. Since M is a power of two, an equality weight separates into a within-block factor and a block factor:

$$\widetilde{\mathsf{eq}}(\mathbf{r}, iM + x) = \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{pos}}, x)\widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{blk}}, i). \tag{42}$$

The root and every recursive level use this convention: every vector, challenge, and norm uses the actual number B of blocks, and $2^{r_{\mathsf{blk}}}$ appears only when evaluating their multilinear extension.

The zero extension in (41) specifies honest commitment formation; an Akita opening does not prove it. Let $\mathsf{Dec}_{\mathsf{shape}}((\mathbf{s}_i)_{i\in[B]})$ be the full decoded commitment-grid vector over the base field, obtained by unpacking every ring element of $\mathbf{G}_{b,M}\mathbf{s}_i$ for $i \in [B]$ and setting only the absent block indices $i \in \{B, \dots, 2^{r_{\mathsf{blk}}} - 1\}$ to zero. Cells of an included final block remain part of this vector, even beyond the declared input length N. Write $\mathsf{Pad}_{\mathsf{shape}}(f)$ for the canonical zero extension of f. Honest commitment satisfies

$$\mathsf{Dec}_{\mathsf{shape}}((\mathbf{s}_i)_i) = \mathsf{Pad}_{\mathsf{shape}}(f),$$

but the knowledge extractor is required only to return the decoded vector on the left. A statement that identifies it with a smaller natural-domain evaluation vector, or with several such vectors embedded into one common grid, must separately establish membership in the image of $\mathsf{Pad}_{\mathsf{shape}}$.

Embedding and decomposition. First pack each run of d_A consecutive base-field coefficients into one element of R_{q,d_A}: direct coefficient packing places them in the monomial positions of (56), and evaluation trace groups them into coordinates over E and applies the trace-packing map ψ of Section 3.1, in the ring-subfield basis. This gives the ring-packed vector $\mathbf{f}$; view it as blocks $(\mathbf{f}_i)_{i\in[B]} \in (R^M_{q,d_A})^B$ under (41). For a full-field source, apply balanced decomposition to its ring coefficients:

$$\mathbf{s}_i := \mathbf{G}^{-1}_{b,M}(\mathbf{f}_i), \qquad \mathbf{G}_{b,M} := \mathbf{I}_M \otimes (1, b, \dots, b^{\delta_{\mathsf{com}}-1}), \qquad \delta_{\mathsf{com}} = \lceil \log_b q \rceil.$$

For a bounded source, including a recursive digit witness or a one-hot input, we may instead use the identity source map: $\mathbf{s}_i := \mathbf{f}_i$, $\delta_{\mathsf{com}} := 1$, and $\mathbf{G}_{b,M} := \mathbf{I}_M$. Its packed-coefficient bound is the envelope of the packed ring coefficients, which can exceed that of the logical digit alphabet under trace packing (22). The schedule accounts for this bound in its response and commitment parameters.

Identity encoding is balanced one-digit decomposition only when every packed coefficient lies in $\mathcal{A}_b = \{-b/2, \dots, b/2 - 1\}$. More generally, a bounded source using balanced decomposition must fit its exact representable interval, with negative reach $(b/2)\sum_{u<\delta_{\mathsf{com}}} b^u$ and positive reach $(b/2-1)\sum_{u<\delta_{\mathsf{com}}} b^u$, after the prescribed lift. A magnitude bound alone does not imply this asymmetric interval condition. In either encoding, $\mathbf{G}_{b,M}\mathbf{s}_i = \mathbf{f}_i$; later fold relations use this recomposition equation.

Inner commitment ($\mathbf{A}$, at d_A). We apply $\mathbf{A}$ to each source block $\mathbf{s}_i$ and decompose the image in balanced base b_1, with depth $\delta_{\mathsf{open}} = \lceil \log_{b_1} q \rceil$:

$$\mathbf{t}_i := \mathbf{A}\,\mathbf{s}_i \in R^{n_A}_{q,d_A}, \qquad \hat{\mathbf{t}}_i := \mathbf{G}^{-1}_{b_1,n_A}(\mathbf{t}_i) \quad (i \in [B]).$$

Sliced outer commitment ($\mathbf{B}$, at d_B). Slicing reuses one outer matrix on consecutive ranges of source blocks, so its width does not grow with the number of blocks; the price is more relation rows. The commitment shape fixes a nonempty power-of-two slice count $S_B \le B$ and derives the canonical block ranges

$$I_{B,s} := \left[\left\lfloor \frac{sB}{S_B} \right\rfloor, \left\lfloor \frac{(s+1)B}{S_B} \right\rfloor \right) \qquad (s \in [S_B]), \tag{43}$$

written $\mathit{\Pi}_B(B, S_B) := (I_{B,s})_{s\in[S_B]}$.

One block of one of the P polynomials fixed by Src contributes

$$g_B := n_A \frac{d_A}{d_B} \delta_{\mathsf{open}}$$

elements of R_{q,d_B} to the outer map. Let $\hat{\mathbf{t}}^{(B)}_{p,i} \in R^{g_B}_{q,d_B}$ be the R_{q,d_B} view of the opening digits of inner image (p, i), ordered by $\mathbf{A}$ row, R_{q,d_B} subcolumn, and opening digit, with the digit coordinate changing first. This view is a fixed permutation of $\hat{\mathbf{t}}_i$ (Appendix L.1); the $\mathbf{A}$ relation retains the original d_A view. Write $\hat{\mathbf{t}}$ for the concatenation in polynomial and then block order. Set

$$L^{\mathsf{blk}}_B := \left\lceil \frac{B}{S_B} \right\rceil, \qquad m_B := PL^{\mathsf{blk}}_B g_B,$$

and form the input vector for slice s as

$$\mathbf{x}_{B,s} := \Big\|_{p\in[P]} \left(\Big\|_{i\in I_{B,s}} \hat{\mathbf{t}}_{p,i}^{(B)} \,\Big\|\, \mathbf{0}^{(L_B^{\mathsf{blk}}-|I_{B,s}|)g_B} \right) \in R_{q,d_B}^{m_B}. \tag{44}$$

A shorter slice is padded inside each polynomial segment. This preserves polynomial-major order and makes the case $S_B = 1$ identical to the unsliced input.

The prover applies one matrix $\mathbf{B} \in R_{q,d_B}^{n_B\times m_B}$ to every slice, $\mathbf{u}_s := \mathbf{B}\mathbf{x}_{B,s} \in R_{q,d_B}^{n_B}$, and stacks the results:

$$\mathbf{u} := (\mathbf{u}_0 \| \cdots \| \mathbf{u}_{S_B-1}) =: \mathbf{B}_{\Pi_B(B,S_B)}\hat{\mathbf{t}}. \tag{45}$$

Its coordinates are ordered by slice, $\mathbf{B}$ row, and ring coefficient, with the coefficient changing first. The right-hand side denotes the block-diagonal map induced by (44), not S_B independently sampled matrices. The complete stack is either transmitted directly or compressed once by the $\mathbf{F}$-chain below, so slicing creates neither separate commitments nor separate payloads. Two certified digit vectors with the same stack differ in some slice, so binding reduces to the single instance $\mathsf{MSIS}_{q,d_B}(n_B, m_B, b^\star - 1)$, where $b^\star$ is the certified range base of the witness carrying these digits (Lemma 4.3). The stacked image has $S_B\, n_B$ ring elements rather than n_B; Appendix L.1 lists the resulting costs.

Commitment compression ($\mathbf{F}$-chain). Sending the outer image $\mathbf{u}$ can be expensive even when the witness being folded is small. We shorten the public message by committing to a binary decomposition of $\mathbf{u}$, and then once more to a decomposition of the resulting image. The intermediate image keeps the input of the second map short, and hence also the output length its security requires. Both maps are sized for their complete inputs under the Module-SIS assumption. At a compressed level both the commitment and opening paths use exactly two maps, $L_F = L_H = 2$; otherwise $L_F = L_H = 0$ and the images are transmitted directly. The schedule fixes this choice and every matrix shape before the commitments are formed.

Both stages use the negative-binary alphabet $\mathcal{A}_{F,j} = \mathcal{A}_{H,j} = \{-1, 0\}$. Write $\delta = \lceil \log_2 q \rceil$. For each $x \in \mathbb{Z}_q$, take the binary expansion of the canonical representative of $-x \bmod q$ and negate its digits, so that

$$x = \sum_{i=0}^{\delta-1} 2^i \xi_i \pmod q, \qquad \xi_i \in \{-1, 0\}.$$

Two valid digit vectors therefore differ coefficientwise by at most one, and the signs let the range-check sum-check certify these digits through the fused constraint $w(w+1) = 0$ (Remark 6.2). Set $\widetilde{\mathbf{u}}_0 := \mathbf{u}$. At each stage $j \in \{1, 2\}$, decompose the preceding image coefficientwise and pack the digits, in the digit-plane order of Appendix L.1, into the input ring elements of $\mathbf{F}_j$. The gadget map $\mathbf{G}_{\mathcal{A}_{F,j},\cdot}$ recomposes each coefficient from its digits, and its inverse denotes the canonical decomposition:

$$\hat{\mathbf{u}}_j := \mathbf{G}^{-1}_{\mathcal{A}_{F,j},\cdot}(\widetilde{\mathbf{u}}_{j-1}), \qquad \widetilde{\mathbf{u}}_j := \mathbf{F}_j\hat{\mathbf{u}}_j.$$

Every coefficient of $\hat{\mathbf{u}}_j$ lies in $\{-1, 0\}$. Only the second image is transmitted; both digit vectors enter the next fold's witness, where the relation check proves the recomposition and commitment equations. The opening image uses the same construction with $\mathbf{H}_1, \mathbf{H}_2$ and digit vectors $\hat{\mathbf{v}}_1, \hat{\mathbf{v}}_2$. The chain indices $j \in \{1, \ldots, L_F\}$ and $j \in \{1, \ldots, L_H\}$ below are empty when the images are transmitted directly. The public payload is

$$\mathbf{u}_{\mathsf{pub}} := \widetilde{\mathbf{u}}_{L_F} = \begin{cases} \mathbf{u}, & L_F = 0, \\ \widetilde{\mathbf{u}}_{L_F}, & L_F > 0, \end{cases} \tag{46}$$

encoded as in Appendix L.1, without a shape or length header; we also write $C := \mathbf{u}_{\mathsf{pub}}$. The compression maps are prefix views of the shared setup (Section 4.3).

With the parameters of Table 10, each complete input image of at most 8 KiB, meaning the concatenation of all $\mathbf{B}$-slice images or, separately, the complete $\mathbf{D}$-image, produces a 256-byte intermediate image and a 128-byte public output. Every such compression instance has at least 147 bits of estimated security in the quantum ADPS16 Core-SVP model (Appendix G). Each distinct commitment has its own complete $\mathbf{B}$-image and pair of compression maps, while all groups of a fold share one $\mathbf{D}$-image (Section 5.4). The small output comes at the cost of two digit vectors in the recursive witness, so the choice between compression and direct transmission weighs the bytes saved now against the work added to later folds (Section 12).

Binding the compression digits. The digit vectors connect the original image to the public payload through successive commitment equations, which become rows of the common relation matrix (Figure 5). To state them once for both commitment paths, let $\mathbf{M}$ be the original map, $(\mathbf{K}_j)_{j\in\{1,\dots,L\}}$ its compression maps, and $\mathbf{p}$ the transmitted payload. The rows assert

$$\mathbf{M}\hat{\mathbf{x}} = \mathbf{G}_{\mathcal{A}_1,\cdot}\hat{\boldsymbol{\xi}}_1, \qquad \mathbf{K}_j\hat{\boldsymbol{\xi}}_j = \mathbf{G}_{\mathcal{A}_{j+1},\cdot}\hat{\boldsymbol{\xi}}_{j+1} \quad (1 \le j < L), \qquad \mathbf{K}_L\hat{\boldsymbol{\xi}}_L = \mathbf{p}. \tag{47}$$

Here $L = 2$ and $\mathcal{A}_1 = \mathcal{A}_2 = \{-1, 0\}$. At depth $L = 0$, the entire chain is replaced by the direct row

$$\mathbf{M}\hat{\mathbf{x}} = \mathbf{p}. \tag{48}$$

For the $\mathbf{F}$ chain, the substitution is

$$(\mathbf{M}, \mathbf{K}_j, \hat{\mathbf{x}}, \hat{\boldsymbol{\xi}}_j, \mathbf{p}) = (\mathbf{B}_{\Pi_B(B,S_B)}, \mathbf{F}_j, \hat{\mathbf{t}}, \hat{\mathbf{u}}_j, \mathbf{u}_{\mathsf{pub}}).$$

For the $\mathbf{H}$ chain, it is

$$(\mathbf{M}, \mathbf{K}_j, \hat{\mathbf{x}}, \hat{\boldsymbol{\xi}}_j, \mathbf{p}) = (\mathbf{D}, \mathbf{H}_j, \hat{\mathbf{e}}, \hat{\mathbf{v}}_j, \mathbf{v}_{\mathsf{pub}}).$$

Only the final payloads $\mathbf{u}_{\mathsf{pub}}$ and $\mathbf{v}_{\mathsf{pub}}$ appear as public commitment targets, each absorbed before the challenges of the fold that checks its chain. The chain therefore binds its compression digits without sending them through the outer commitment pipeline again, which would make the commitment input depend on digits derived from a commitment to that same input. The range check at the common balanced alphabet does not by itself justify the compression maps' collision bound of one; the fused constraint $w(w + 1) = 0$ supplies it. The deterministic honest decomposition supplies a canonical witness, while the binary constraint bounds the openings of a cheating prover.

Lemma 4.3 (Compression binding chain). *Fix a depth $L \in \{0, 2\}$, a public payload $\mathbf{p}$, and either compression rows of the form (47) when $L > 0$ or the uncompressed row (48) when $L = 0$. Suppose two tuples $(\hat{\mathbf{x}}, (\hat{\boldsymbol{\xi}}_j)_{j\in\{1,\dots,L\}})$ and $(\hat{\mathbf{x}}', (\hat{\boldsymbol{\xi}}'_j)_{j\in\{1,\dots,L\}})$ satisfy the selected rows with the same $\mathbf{p}$, with $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$ in a matrix-input alphabet $\mathcal{A}_{\mathsf{in}}$ certified by the range check, and with every coefficient of both $\hat{\boldsymbol{\xi}}_j$ and $\hat{\boldsymbol{\xi}}'_j$ in $\{-1, 0\}$ for every j, as certified by the fused binariness check on its complete digit span. Let $\mathbf{M}_{\mathsf{full}} := \mathbf{M}$ for an unsliced map and $\mathbf{M}_{\mathsf{full}} := \mathbf{B}$ for the sliced outer map $\mathbf{M} = \mathbf{B}_{\Pi_B(B,S_B)}$. Write $\mathbf{M}_{\mathsf{full}} \in R_{q,d_M}^{n_{\mathsf{full}}\times m_{\mathsf{full}}}$; in the sliced case, $(n_{\mathsf{full}}, m_{\mathsf{full}}) = (n_B, m_B)$. If the two tuples differ, then one efficiently extracts one of:*
1. a solution to the matrix instance $\mathsf{MSIS}_{q,d_M}(n_{\mathsf{full}}, m_{\mathsf{full}}, \eta_{\mathsf{in}})$ for $\mathbf{M}_{\mathsf{full}}$, where $\eta_{\mathsf{in}} := \max_{a,a'\in\mathcal{A}_{\mathsf{in}}} |a - a'|$;
2. for some $j \in \{1, \dots, L\}$, a solution to $\mathsf{MSIS}_{q,d_{K,j}}(n_{K,j}, |\hat{\boldsymbol{\xi}}_j|, 1)$ for $\mathbf{K}_j$.

Proof. If $L = 0$, subtracting the two uncompressed rows gives $\mathbf{M}(\hat{\mathbf{x}} - \hat{\mathbf{x}}') = 0$. Since the tuples differ, this is a nonzero collision with the certified matrix-input norm.

Suppose now that $L > 0$. Walk backward from the payload and let j be the largest index at which the two chain digit vectors differ. The corresponding row gives $\mathbf{K}_j(\hat{\boldsymbol{\xi}}_j - \hat{\boldsymbol{\xi}}'_j) = 0$. Its coefficient norm is at most one, since both digit vectors have coefficients in $\{-1, 0\}$. If every chain digit vector agrees, then the first row gives $\mathbf{M}(\hat{\mathbf{x}} - \hat{\mathbf{x}}') = 0$; since the tuples differ, the matrix-input difference is nonzero and has norm at most η_{in}. When the matrix is sliced, the kernel vector restricts to some nonzero slice. Padding that restriction with zeros to the scheduled width gives a nonzero kernel vector of $\mathbf{B}$ itself with the same norm. The same argument applied directly to the unsliced matrix $\mathbf{D}$ proves binding for the opening chain.

4.6 Commitments Formed before the Fold

To admit commitments formed before their batch is known, the *commitment contract*, the public data fixed when the commitments are formed, includes a common certified root alphabet with $b^\star = 2^{l^\star}$. A carried digit base $b_{1,g} > b^\star$ could make an honest opening unencodable; when $b_{1,g} < b^\star$, binding must still use the larger alphabet that the range check enforces. Choosing that alphabet only at opening time cannot repair a matrix rank fixed earlier. The opening descriptor of each group (Section 5.1) also requires $b_{z,g} \le b^\star$ for response digits. Neither base need equal the source-recomposition base b_g. Rather than separate range predicates for different alphabets, we derive each map's collision bound from the common digit diameter.

Commitment pipeline (one commitment).

$$f \xrightarrow{\text{embed into } R_{q,d_A}} \text{ring-packed vector } \mathbf{f} \xrightarrow{\text{reshape}} (\mathbf{f}_i)_{i\in[B]}$$
$$\xrightarrow{\mathbf{G}_{b,M}^{-1}} (\mathbf{s}_i) \xrightarrow{\mathbf{A}} (\mathbf{t}_i) \xrightarrow{\mathbf{G}_{b_1}^{-1}} \hat{\mathbf{t}}$$
$$\hat{\mathbf{t}} \xrightarrow{\mathbf{B} \text{ on canonical slices}} \mathbf{u} \xrightarrow{\text{optional } \mathbf{F}\text{-chain}} \underbrace{\mathbf{u}_{\mathsf{pub}}}_{\text{payload}}$$

Ring dimensions per arrow: $\mathbf{A}$ at d_A, $\mathbf{B}$ at d_B, and the $\mathbf{F}$-chain at $(d_{F,j})_{j\in\{1,\dots,L_F\}}$. When the chain is active, its digit vectors $(\hat{\mathbf{u}}_j)_{j\in\{1,\dots,L_F\}}$ join the next recursive witness and only $\mathbf{u}_{\mathsf{pub}}$ is transmitted; at depth zero, the payload is the outer image $\mathbf{u}$. The opening path $\hat{\mathbf{e}} \to \mathbf{D} \to$ optional $\mathbf{H}$-chain $\to \mathbf{v}_{\mathsf{pub}}$ has the same form (Figure 4).

Fig. 3: The commitment data path. The transmitted payload is either the outer Ajtai image or the terminal image of its scheduled compression chain. All hidden intermediate images are bound by relation rows (Section 7).

For group g, the root range check certifies the balanced base-$b^\star$ alphabet. Two accepted outer digit vectors therefore differ coefficientwise by at most

$$\eta_{B,g} = b^\star - 1. \tag{49}$$

At opening time, the opening descriptor of group g fixes its challenge family, response base $b_{z,g}$, response depth $\delta_{f,g}$, and challenge-difference bound $\bar{\kappa}_{1,g}$. These data determine the exact inner collision norm

$$\Delta_{f,g}^{\mathsf{cert}} = (b^\star - 1)\frac{b_{z,g}^{\delta_{f,g}} - 1}{b_{z,g} - 1}, \qquad \eta_{A,g} = 2\bar{\kappa}_{1,g}\Delta_{f,g}^{\mathsf{cert}}. \tag{50}$$

Neither norm depends on the other groups or their points. The verifier admits the descriptor only when the fixed ranks and security contract of the pre-existing commitment cover these exact bounds. The fold may choose a compatible recursive suffix and prover partition, but cannot resize or reinterpret an incoming commitment.

5 One Fold for an Opening Batch

Our commitments are now fixed. We check their opening claims together and continue with one recursive witness. As in Greyhound [82, §3.2], each opening group keeps its own response and its own inner and sliced outer commitment path. All partial opening evaluations share one unsliced $\mathbf{D}/\mathbf{H}$ path, and the responses and auxiliary values are checked in one relation and one digit-range argument. Every group uses the opening method prescribed for the fold's absolute level, though the groups' dimensions and points may differ.

Throughout this section, h denotes the fold level, g an opening group, c a claim in that group, and i a source block. Inside one group we suppress its group and claim indices. The subsections below define the objects of the fold; Figure 4 gives the message order.

5.1 Inputs and Parameters

Fix an ordered list of G opening groups. Group $g \in [G]$ references a frozen commitment handle from Section 4.2 and one opening point $\mathbf{r}_g$. It contains the $n_{\mathsf{clm},g} = P_g$ Boolean evaluation vectors $F_{g,c}$, $c \in [n_{\mathsf{clm},g}]$, of that commitment's ordered source layout. A public projection $\pi_{g,c}$ selects the coordinates used by $\widetilde{F_{g,c}}$, and $v_{g,c}$ is the claimed value. The local claims assert

$$\widetilde{F_{g,c}}(\pi_{g,c}(\mathbf{r}_g)) = v_{g,c} \qquad \text{for } c \in [n_{\mathsf{clm},g}].$$

The projections let polynomials of different multilinear dimensions share an instance. When admitting the statement, the verifier checks that each $\pi_{g,c}$ agrees with the coordinate projection induced by that vector's frozen padding embedding. To open a commitment at another point, another group references the same handle. Duplicate opening-group requests are removed before the root opening layout is derived; this removes no vector from a commitment and changes neither its Src nor P_g.

Putting local claims on one grid. The honest commitment algorithm zero-pads the vectors of group g to a common μ_g-variable grid. Let $\phi_{g,c}$ be the product of the equality factors on the padded coordinates. The padded vector $F^{\mathsf{pad}}_{g,c} := \mathsf{Pad}_{g,c}(F_{g,c})$ satisfies

$$\widetilde{F^{\mathsf{pad}}_{g,c}}(\mathbf{r}_g) = \phi_{g,c}\,\widetilde{F_{g,c}}(\pi_{g,c}(\mathbf{r}_g)). \tag{51}$$

The verifier derives $\phi_{g,c}$ from the public layout and rejects if it is zero. This is a public admission condition, not a negligible event for an arbitrary caller-chosen point. A claim with $\phi_{g,c} = 0$ must stay in a separate natural-dimension instance. The batching layer therefore supports exactly the admitted layouts, not every point-and-dimension combination of the base PCS API.

The padding identity describes honest encodings only. For a malicious commitment, extraction returns a decoded common-grid vector $F^{\star}_{g,c}$ with $\widetilde{F^{\star}_{g,c}}(\mathbf{r}_g) = \phi_{g,c}v_{g,c}$, but not $F^{\star}_{g,c} \in \mathrm{im}(\mathsf{Pad}_{g,c})$. An application that reads this as the natural-domain claim must certify that image condition separately, or use a commitment shape whose common grid is the vector's natural domain. The batch knowledge-soundness theorem of Section 10.3 uses this convention.

Group geometry. Group g has A-ring dimension $d_{A,g}$ and power-of-two block length M_g, both fixed by the incoming commitment. If each padded vector's ring-packed representation has N_g ring elements, the number of nonempty blocks is

$$B_g := \left\lceil \frac{N_g}{M_g} \right\rceil. \tag{52}$$

For $i \in [B_g]$, let $\mathbf{f}_{g,c,i} \in R_{q,d_{A,g}}^{M_g}$ be block i of the ring-packed representation of $F^{\mathsf{pad}}_{g,c}$, formed by the honest commitment algorithm as in (41). Distinct groups may use different A-ring dimensions, challenge-subring dimensions, packing factors, and challenge families.

With coefficient packing, the opening descriptor of group g (Section 5.1) fixes a challenge-subring dimension $d_{\mathsf{car},g}$. The packing geometry and embedding of Section 5.3 then hold per group, with $h_{\mathsf{car},g} := d_{A,g}/(kd_{\mathsf{car},g})$ and $\iota_g(c(Y)) := c(X^{kh_{\mathsf{car},g}})$. The descriptor is admissible only when $kd_{\mathsf{car},g} \mid d_{A,g}$, an audited challenge family exists at $d_{\mathsf{car},g}$, and its scheduled coordinate permutation places the position and low-coefficient axes among coordinates shared by every local projection. All claims of the group then use the same contraction $\mathcal{L}_g$ of Section 5.3; if no such common layout exists, the claims must be placed in separate opening instances. The common point segments are $\mathbf{r}_{\mathsf{pos},g}$ and $\mathbf{r}_{\mathsf{pack},g}$. The claim-local segments $\mathbf{r}_{\mathsf{tail},g,c}$ and $\mathbf{r}_{\mathsf{blk},g,c}$ retain the projection and padding weights. Their lengths are fixed by $(M_g, k, h_{\mathsf{car},g}, d_{\mathsf{car},g}, B_g)$; callers do not choose a second coefficient layout. The padded claim $\phi_{g,c}v_{g,c}$ is then the direct scalar opening (59) of group g at these segments, with the coordinate planes $e_{g,c,i,t,u}$ of (58).

With evaluation trace, the descriptor instead fixes the shared tensor-reduction basis split, the group's native tail arity, and the full-ring challenge family at $d_{A,g}$. Each padded claim passes through the shared tensor reduction of Section 5.2 and is then opened by the trace row (98).

Fold description. Before any challenge, the fold binds compatible response parameters. The *opening descriptor* of each group contains its commitment reference, claim projections, absolute level, and the lengths and alphabets of its distinguished witness segments. A packing descriptor adds its challenge-subring geometry; a trace descriptor adds its tensor split and trace-packing geometry. For an ordered incoming opening shape $\mathcal{O}_h$, the fold description extends the configuration (31) by per-group challenge and response data and the shared opening path:

$$\mathsf{Fold} := \big(\mathsf{cfg}_h;\ (\mathsf{Chal}_g, b_{z,g}, \delta_{f,g})_{g\in[G]};\ (d_D, n_D);\ \boldsymbol{\delta}_{\mathsf{quot}}\big). \tag{53}$$

The fields are as follows.

– Open_h is the opening method derived from (127).

- $\mathsf{RingCheck}_h$ selects $\mathsf{QuotientLift}$ or $\mathsf{QuotientFree}$. The vector $\boldsymbol{\delta}_{\mathsf{quot}}$ records the quotient digit depths in the former case and is empty in the latter (Section 7).
- Comp_h collects the source and target $\mathcal{F}$ profiles and the shared profile $\mathcal{H}$, which together with (d_D, n_D) specifies the unsliced shared opening path. As for $\mathcal{F}$ in (28), $\mathcal{H}$ is empty or contains the two triples $(\{-1, 0\}, d_{H,i}, n_{H,i})$ for $i \in \{1, 2\}$. Every source path and the shared path use the same compression mode; the target profile specifies its successor's mode.
- Norm_h selects a coefficient-wise bound or an additional squared-norm certificate. Packing folds use the coefficient-wise route. For simplicity, the squared-norm route is used only for a single recursive response after the setup openings have been checked and response chunking has ended (Definition 9.2). It records the squared-norm bound, the challenge operator bound, and the exact norm-proof domain of Section 6.2.
- Chal_g specifies the base family, every filter, a certified lower bound on the accepted-family cardinality, the draw limit, the challenge family for the source blocks, and the response acceptance rule. With coefficient packing, the accepted family lies in the challenge subring and acts in the A ring through (38); with evaluation trace, it lies in the full A ring.
- $b_{z,g}$ and $\delta_{f,g}$ fix the positional base and digit depth of the group's response. The incoming commitment's base b_g still reconstructs the source blocks; these bases need not coincide.

The shared nonce set, of size $N_{\mathsf{try},h}$, and the joint retry rule belong to the level description (Section 4.2). An evaluation-trace fold also records the tensor-reduction shape that converts its incoming ordinary opening before the fold challenge. The target shape $\mathcal{O}'$ records the successor commitment and ordered witness segments. Together these records determine the block counts, matrix widths, compression widths, and successor length through Section 4.5 and (69). Collision bounds and induced Module-SIS instances are derived from them.

5.2 Shared Tensor Reduction

With evaluation trace, the tensor reduction of Section 3.7 extends to several opening groups without moving their claims to a common point. For each claim, let $\bar{f}_{g,c}$ be the packed polynomial of $F^{\mathsf{pad}}_{g,c}$ in the fixed extension-field basis. Its input claim is $\phi_{g,c} v_{g,c}$ at $\mathbf{r}_g$, so the padding factor is part of the reduction input and is not multiplied again in the evaluation-trace scalar row. Write ℓ_g for the arity of group g's base-field vectors, $m_g := \ell_g - \kappa$ for its tail arity, and $A_{\eta,g}$ for the batched equality factor A_η formed from its point. All groups use the same extension basis and split parameter κ, but their points and tail arities may differ. Set $m_{\max} := \max_g m_g$.

To use one sum-check across different arities, define, for $w \in \{0,1\}^{m_g}$ and $z \in \{0,1\}^{m_{\max}-m_g}$, the cylindrical extensions

$$\widehat{\bar{f}}_{g,c}(w, z) := \bar{f}_{g,c}(w), \qquad \widehat{A}_{\eta,g}(w, z) := A_{\eta,g}(w)\widetilde{\mathsf{eq}}(\mathbf{0}, z).$$

The witness does not depend on the extra coordinates, while the equality factor restricts them to zero. Hence the extended Boolean sum equals the native tensor-reduction claim for group g. The extension changes only the sum-check expression; it does not pad the group's committed vector.

Write $N_{\mathsf{tensor}} := \sum_g n_{\mathsf{clm},g}$ for the number of claims. On a nonidentity reduction, after binding every opening value and column partial, the verifier samples the shared row-batching point η and normalized claim-batching coefficients $(\zeta^{\mathsf{tensor}}_{g,c})$: the first is one, and the remaining $N_{\mathsf{tensor}} - 1$ are independent samples from E. It then runs one $m_{\max}$-round degree-2 sum-check for

$$\sum_g \sum_{c \in [n_{\mathsf{clm},g}]} \zeta^{\mathsf{tensor}}_{g,c} \widehat{\bar{f}}_{g,c}(x) \widehat{A}_{\eta,g}(x). \tag{54}$$

Let $\rho \in E^{m_{\max}}$ be the final sum-check point and let $\rho_g := \rho_{[m_g]}$ be its prefix of length m_g. The transparent factor for group g is

$$\theta_g := A_{\eta,g}(\rho_g) \prod_{t=m_g}^{m_{\max}-1} (1 - \rho_t).$$

The prover supplies one final product $\tilde{v}_{g,c} := \theta_g \bar{f}_{g,c}(\rho_g)$ for every claim. The verifier checks that the sum-check final value equals $\sum_{g,c} \zeta^{\mathsf{tensor}}_{g,c} \tilde{v}_{g,c}$ and rejects if any θ_g is zero. Each input point stays bound separately,

and each packed claim $\bar{f}_{g,c}(\rho_g)$ uses the prefix and packing shape of its own group. An identity reduction sets $\theta_g = 1$ and forwards the padded input claims under their canonical packing identification. For $k = 1$, this means $\rho_g = \mathbf{r}_g$ and $\tilde{v}_{g,c} = \phi_{g,c} v_{g,c}$, with no tensor-prefix challenges.

Theorem 5.1 (Shared tensor-reduction soundness). *Consider a nonidentity shared tensor reduction of N_{tensor} claims in G groups over $E = \mathbb{F}_{q^{2^\kappa}}$.*

1. (Soundness.) *If at least one input claim is false, then the probability that all tensor-reduction checks pass and every forwarded packed claim is correct is at most*
$$\frac{\kappa + \mathbf{1}_{\{N_{\mathsf{tensor}} > 1\}} + 2m_{\max}}{|E|}.$$
2. (Honest abort.) *For an honest prover, the probability that at least one of the G group factors θ_g vanishes is at most*
$$\frac{G(\kappa + m_{\max})}{|E|}.$$

An identity reduction omits the row-batching, claim-batching, and sum-check challenges, and both of its errors are zero.

The proof, and the extraction parameters used by the knowledge-soundness analysis, are in Appendix D.1.

5.3 Forming the Partial Opening Evaluations

The committed polynomial is defined over $K = \mathbb{F}_q$, while its opening point lies in E. Its base-field Boolean evaluations are packed into ring elements, and folding multiplies their blocks by ring challenges. We need partial opening evaluations that recover the claimed field value and remain consistent with this ring multiplication. Direct coefficient packing evaluates some coefficient coordinates early and uses subring challenges. Evaluation trace uses a field embedding and trace-compatible packing to permit full-ring challenges.

Direct Coefficient Packing Direct coefficient packing places input values directly in the coefficient positions of the commitment ring. We split each coefficient index into two parts: one is evaluated at the original opening point, while the other indexes the coefficients of a smaller polynomial over E. Subring challenges act on this second index without mixing the first, so evaluating before folding gives the same result as folding before evaluating.

Packing geometry. The schedule selects the A-ring dimension d_A and the challenge-subring dimension d_{car} of Section 4.4, with
$$d_A = k h_{\mathsf{car}}\, d_{\mathsf{car}}, \qquad h_{\mathsf{car}} \in \mathbb{Z}_{\geq 1}, \tag{55}$$
where $k, d_A, h_{\mathsf{car}}, d_{\mathsf{car}}$ are powers of two in the supported parameter sets. We use the A ring R, the challenge subring S, the extension-field carrier ring S_E of (37), and the embedding $\iota : S \hookrightarrow R$, $Y \mapsto X^{k h_{\mathsf{car}}}$, of (38). Each partial opening evaluation lies in S_E; each of its coefficients lies in E and needs k base-field coordinates, so a partial is h_{car} times narrower than an element of R in base-field width.

Canonical coefficient split. Use the block geometry of (40) and (41), with $i \in [B]$ and $x \in [M]$. Write the element of R at that address as
$$F_{i,x}(X) = \sum_{j \in [d_{\mathsf{car}}]} \sum_{a \in [k h_{\mathsf{car}}]} f_{i,x,a,j} X^{a + k h_{\mathsf{car}} j}, \qquad f_{i,x,a,j} \in K. \tag{56}$$

The coefficient positions form consecutive groups of size $k h_{\mathsf{car}}$: a indexes a position within group j, giving ring-coefficient index $a + k h_{\mathsf{car}} j$. The matching part of the opening point is split as $\mathbf{r}_{\mathsf{pack}} \in E^{\log(k h_{\mathsf{car}})}$ and $\mathbf{r}_{\mathsf{tail}} \in E^{\log d_{\mathsf{car}}}$. The first point contracts a; the second is retained until the scalar opening equation below.

For every source block, the prover contracts the position and low-coefficient axes to obtain one element of the extension-field carrier ring:
$$e_i(Y) := \sum_{x \in [M]} \sum_{a \in [k h_{\mathsf{car}}]} \sum_{j \in [d_{\mathsf{car}}]} \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{pos}}, x) \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{pack}}, a) f_{i,x,a,j} Y^j \in S_E. \tag{57}$$

Writing

$$e_i(Y) = \sum_{j\in[d_{\mathsf{car}}]} \left(\sum_{t\in[k]} \beta_t e_{i,t,j} \right) Y^j, \qquad e_{i,t,j} \in K, \tag{58}$$

shows that one partial contains exactly kd_{car} committed base-field coordinates. The canonical coordinate order is block, extension coordinate t, and carrier coordinate j, preceded by the claim index when several polynomials share a group.

Direct scalar opening. Contracting the block and carrier axes of these partials recovers the original extension-field claim. For one polynomial, the required equation is

$$\sum_{i\in[B]} \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{blk}}, i) \sum_{j\in[d_{\mathsf{car}}]} \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{tail}}, j) \sum_{t\in[k]} \beta_t e_{i,t,j} = v. \tag{59}$$

This is one E-valued linear row. It uses neither a trace map nor a tensor-reduction sum-check, and Section 7.1 checks it on the opening digits.

One challenge in two rings. Once every coordinate plane of the partials has been bound, the verifier samples each block challenge from an audited family $\mathcal{C}_{d_{\mathsf{car}}} \subset S$. The same challenge acts in two rings:

$$\begin{aligned} c_i(Y) &\in S \quad \text{in the carrier relation,} \\ \iota(c_i) = c_i(X^{kh_{\mathsf{car}}}) &\in R \quad \text{in the inner-commitment relation.} \end{aligned} \tag{60}$$

No independent A-ring challenge is sampled. If $c - c'$ is a unit in S, then $\iota(c - c')$ is a unit in R (Lemma 4.2), so the special-soundness argument may use the same fork in both relations.

The same challenge can fold both the partials and the source blocks. Let $\mathcal{L}$ denote the contraction in (57), so that $\mathcal{L}(\mathbf{f}_i) = e_i$. The coefficient split makes $\mathcal{L}$ linear over S:

$$\mathcal{L}\big(\iota(c)F\big) = c\,\mathcal{L}(F) \qquad \text{in } S_E. \tag{61}$$

Indeed, multiplication by Y advances the carrier index j, while a wrap at $j = d_{\mathsf{car}}$ contributes the same minus sign as a wrap at X^{d_A}. Consequently, for the folded source

$$Z_x(X) := \sum_{i\in[B]} \iota(c_i) F_{i,x}(X), \tag{62}$$

the native consistency relation is

$$\mathcal{L}(Z)(Y) = \sum_{i\in[B]} c_i(Y) e_i(Y) \qquad \text{in } S_E. \tag{63}$$

The inner-commitment rows remain over R and use $\iota(c_i)$:

$$[\mathbf{A}\mathbf{G}_{b_z,m_A}\hat{\mathbf{z}}]_r = \sum_{i\in[B]} \iota(c_i)[\mathbf{G}_{b_1,n_A}\hat{\mathbf{t}}_i]_r \qquad (r \in [n_A]). \tag{64}$$

Thus only the consistency row moves to the carrier ring; the matrix $\mathbf{A}$ and its setup remain over R.

Base-field specialization. When $k = 1$, the planner may still take $d_{\mathsf{car}} < d_A$ to shorten the partial and carrier quotient. Taking $d_{\mathsf{car}} = d_A$ recovers the ordinary full-ring opening.

Evaluation Trace For evaluation trace, we first use the shared tensor reduction of Section 5.2 to obtain packed claims at their native points. We again suppress the group and claim indices. Let ρ be this group's packed opening point. The schedule splits this point according to the trace layout as

$$\rho = (\rho_{\mathsf{blk}}, \rho_{\mathsf{pos}}, \rho_{\mathsf{pack}}), \tag{65}$$

for the block, within-block position, and trace-packing axes. For $x \in [M]$, set $a_x := \widetilde{\mathsf{eq}}(\rho_{\mathsf{pos}}, x) \in E \subseteq R_{q,d_A}$ and $\mathbf{a} := (a_x)_{x\in[M]}$. The evaluation-trace method then produces the ordinary full-ring partial

$$e_i := \langle \mathbf{a}, \mathbf{f}_i \rangle \in R_{q,d_A}, \tag{66}$$

with d_A base-field coordinates.

5.4 Binding the Partials and Folding the Sources

Reusing the incoming commitments. For group g, apply Section 4.5 with $P = n_{\mathsf{clm},g}$ and $B = B_g$, adding indices (g, c) to the local source blocks and inner images. Thus $\mathbf{s}_{g,c,i}$, $\mathbf{t}_{g,c,i} = \mathbf{A}_g \mathbf{s}_{g,c,i}$, and their base-$b_{1,g}$ digits $\hat{\mathbf{t}}_{g,c,i}$ are already committed. The slice width, the slice inputs $\mathbf{x}_{B,g,s}$ of (44), and the images $\mathbf{u}_{g,s} = \mathbf{B}_g \mathbf{x}_{B,g,s}$ keep the handle's frozen padding and compression path. A group that opens the same commitment at another point reuses these data.

Binding the partials. Each group's partials depend on its own point, but one opening commitment binds all of them. At a packing level, for each local claim and source block, the prover applies the group-local contraction $\mathcal{L}_g$ of (57), using $\mathbf{r}_{\mathsf{pos},g}$ and $\mathbf{r}_{\mathsf{pack},g}$:

$$e_{g,c,i}(Y) = \sum_{u \in [d_{\mathsf{car},g}]} \left(\sum_{t \in [k]} \beta_t e_{g,c,i,t,u} \right) Y^u \in E[Y]/(Y^{d_{\mathsf{car},g}} + 1). \tag{67}$$

At an evaluation-trace level, each group instead uses the full-ring partials (66) of its tensor-reduction output. One partial of group g has base-field width

$$d_{\mathsf{op},g} := \begin{cases} k d_{\mathsf{car},g} & \text{with coefficient packing,} \\ d_{A,g} & \text{with evaluation trace.} \end{cases} \tag{68}$$

Its coefficients, in layout $[g, c, i, t, u]$ with packing or $[g, c, i, u]$ with evaluation trace, are split into native d_D-coefficient blocks and gadget-decomposed in base $b_{1,g}$ (Section 5.5). Let $\hat{\mathbf{e}}_g$ be the resulting interval for group g, and let $\hat{\mathbf{e}} := \mathsf{concat}(\hat{\mathbf{e}}_0, \ldots, \hat{\mathbf{e}}_{G-1})$ in the fixed input-group order. The intervals may have different lengths; we concatenate them without padding to a common geometry.

One unsliced opening matrix commits to the whole vector. The fold selects a shared compression depth $L_H \in \{0, 2\}$, with $L_{F,g} = L_H$ for every g and $L_H = 2$ at the root, and shared d_D and n_D such that d_D divides every $d_{\mathsf{op},g}$. Group g contributes

$$m_{D,g} := n_{\mathsf{clm},g} B_g \delta_{\mathsf{open},g} \frac{d_{\mathsf{op},g}}{d_D} \tag{69}$$

columns over R_{q,d_D}. Set

$$m_D := \sum_{g \in [G]} m_{D,g}, \qquad \mathbf{D} \in R_{q,d_D}^{n_D \times m_D}, \qquad \mathbf{v} := \mathbf{D}\hat{\mathbf{e}}. \tag{70}$$

If $L_H = 0$, the public opening payload is $\mathbf{v}_{\mathsf{pub}} = \mathbf{v}$; otherwise the compression chain $(\mathbf{H}_a)_{a \in \{1,\ldots,L_H\}}$ starts from $\mathbf{v}$ and terminates at that payload. The verifier checks n_D against this exact total width and the certified difference bound of the heterogeneous digit vector.

Combining the scalar claims. Akita combines the groups' scalar claims only after the shared opening payload binds the partials; the security proof uses only that a weak opening determines its partials from the committed source (Section 5.4). Write $\Lambda := \sum_{g \in [G]} n_{\mathsf{clm},g}$ and number the claims in group, then claim order, starting at zero.

At a packing root with $\Lambda > 1$, we sample $\zeta_{\mathsf{batch}} \leftarrow E$ and assign successive powers to the claims. If $g(a), c(a)$ identify claim a, its weight is

$$\vartheta_{g(a),c(a)} := \zeta_{\mathsf{batch}}^a, \qquad a \in [\Lambda]. \tag{71}$$

At an offloaded boundary under packing, the ordered setup and witness groups each contain one claim; we use weights $(1, \vartheta_{\mathsf{grp}})$ for a fresh $\vartheta_{\mathsf{grp}} \leftarrow E$. At a singleton direct edge, the only weight is one. The packing target is the corresponding sum of padded values:

$$v_{\mathsf{root}} := \sum_{g \in [G]} \sum_{c \in [n_{\mathsf{clm},g}]} \vartheta_{g,c} \phi_{g,c} v_{g,c}. \tag{72}$$

We keep the subscript root even when the fold receives an offloaded setup opening; at an application root this target is also denoted v_{batch}. At an evaluation-trace fold, after the payload is bound, the first scalar coefficient is one and the other $\Lambda - 1$ are independent samples from E, separate from the claim-batching coefficients inside the tensor reduction. The trace target is $\bar{v}_{\mathsf{root}} := \sum_{g,c} \vartheta_{g,c} \tilde{v}_{g,c}$.

Opening under a supplied row. An application may supply the root row instead. Before the root fold challenge it fixes public weights $\vartheta_{g,c}$ and a public target v_{root}, possibly computed from its own challenges, in place of (71) and (72); the fold is otherwise unchanged. Proposition 10.44 shows that the opening then proves this row for the committed vectors, and Theorem 10.45 states what the application must prove before the row. Neither result uses the order of the opening payload and the weights, so an application may fix its weights before the payload.

Fold challenges and responses. After the scalar weights are fixed, the verifier samples one challenge $c_{g,c,i}$ for every group, local claim, and source block. With coefficient packing, $c_{g,c,i} \in \mathcal{C}_{d_{\mathsf{car},g}}$ and $c^A_{g,c,i} = \iota_g(c_{g,c,i})$; with evaluation trace, $c_{g,c,i} = c^A_{g,c,i} \in \mathcal{C}_{d_{A,g}}$ is a full-ring challenge. A single nonce selects the domain-separated tapes of all coordinate samplers. If every rejection sampler succeeds, the resulting tuple is uniform over the product of the scheduled accepted families in the random-oracle model. The prover keeps the tuple only when every group response

$$\mathbf{z}_g := \sum_{c\in[n_{\mathsf{clm},g}]} \sum_{i\in[B_g]} c^A_{g,c,i}\mathbf{s}_{g,c,i} \tag{73}$$

meets that group's scheduled response certificate. A retry resamples the whole tuple, and the verifier enforces the common nonce limit. With the response base $b_{z,g}$ and depth $\delta_{f,g}$ of group g, the prover decomposes

$$\hat{\mathbf{z}}_g := \mathbf{G}^{-1}_{b_{z,g},m_{A,g}}(\mathbf{z}_g), \qquad \mathbf{z}_g = \mathbf{G}_{b_{z,g},m_{A,g}}\hat{\mathbf{z}}_g. \tag{74}$$

5.5 The Witness Passed to the Next Level

To check how the responses were obtained, the next witness also carries the partial opening evaluations, inner images, and compression digits. First collect the response and auxiliary digits into

$$\mathbf{w}^{(h)}_{\mathsf{main}} := \Big\|_{g\in[G]} [\hat{\mathbf{z}}_g \mid \hat{\mathbf{e}}_g \mid \hat{\mathbf{t}}_g], \qquad \hat{\mathbf{c}}_a := \left(\Big\|_{g\in[G]} \hat{\mathbf{u}}_{g,a} \right) \| \hat{\mathbf{v}}_a. \tag{75}$$

Here $\hat{\mathbf{c}}_a$ contains every group's $\mathbf{F}_a$ digits, then the shared $\mathbf{H}_a$ digits. With response chunking, the main vector instead collects the response and auxiliary digits for each group within each chunk, then concatenates the chunks in order (Section 11.2).

The bases $b_{z,g}$ and $b_{1,g}$ reconstruct the response and opening values, respectively. The one range check certifies every carried digit in the common balanced base-$b^\star$ alphabet fixed by the public commitment contract of Section 4.6, so admission requires every carried digit base, including $b_{z,g}$, to be at most $b^\star$. Compression digits additionally satisfy $w(w+1) = 0$, which restricts them to $\{-1, 0\}$ (Remarks 6.2 and 10.24).

Native ring blocks and digit order. Each witness segment keeps its own ring dimension: response digits use d_A coefficients per ring element, and partial-opening and inner-image digits use d_D and d_B. A value is split into consecutive native blocks before digit decomposition, so the coefficient index changes first, then the digit, then the native block. Appendix L.2 gives the exact addresses.

The complete witness for the next level. On the quotient-lift route, write $\hat{\mathbf{Q}}_{\mathsf{main}}$ for the quotient digits of the fold-evaluation, inner-consistency, outer, and opening relations (families 1–4 of Figure 5): for each group, its fold-evaluation, $\mathbf{A}$, and sliced $\mathbf{B}$ rows, then the shared $\mathbf{D}$ rows. Write $\hat{\mathbf{Q}}_a$ for the compression quotients at depth a, with all group-local $\mathbf{F}_a$ rows before the shared $\mathbf{H}_a$ rows. For $L_{\mathsf{comp}} := L_{F,g} = L_H \in \{0, 2\}$, the successor order is

$$\mathbf{w}^{(h+1)} := \mathbf{w}^{(h)}_{\mathsf{main}} \| \hat{\mathbf{Q}}_{\mathsf{main}} \Big\| \Big\|_{a=1}^{L_{\mathsf{comp}}} [\hat{\mathbf{c}}_a \mid \hat{\mathbf{Q}}_a]. \tag{76}$$

Every $\hat{\mathbf{Q}}$ segment is empty on the quotient-free route; at compression depth zero, the entire compression suffix is absent. Quotients keep their row's native dimension and coefficient field, and the range check covers their digits in the common balanced alphabet. Equation (76) suppresses the scheduled zero alignment between segments and the final zero suffix that extends the vector to the Boolean domain. These zeros are part of the committed vector; Appendix L.2 specifies them.

Certified response envelopes. Suppress the group index and write $b := b_{z,g} \le b^\star$. The group stores its response as δ_f balanced base-b digit planes. The schedule chooses δ_f so that the honest response is encodable by δ_f canonical digits (Appendix K.5). The next level certifies only that each transmitted digit lies in the possibly wider balanced base-$b^\star$ alphabet. Recomposition of an arbitrary accepted digit vector therefore gives the verifier-enforced coefficient envelope

$$\beta_{\mathsf{fold}}^{\mathsf{cert}} = \frac{b^\star}{2} \cdot \frac{b^{\delta_f} - 1}{b - 1}. \tag{77}$$

This bound applies to every accepted response and determines the Module-SIS collision radius, independently of how δ_f was chosen. Weak binding uses the difference of two accepted responses. Two digits in $\{-b^\star/2, \dots, b^\star/2 - 1\}$ differ by at most $b^\star - 1$, so the corresponding certified difference envelope is

$$\Delta_f^{\mathsf{cert}} = (b^\star - 1) \cdot \frac{b^{\delta_f} - 1}{b - 1}. \tag{78}$$

On the coefficient route, the schedule combines Δ_f^{cert} with the certified challenge-difference bound to derive the $\mathbf{A}$-collision radius in (50), which it supplies to the Module-SIS estimator. On the Euclidean route, the same digit ranges remain in force and the next level additionally certifies $\mathcal{E}_{\mathsf{int}}(\hat{\mathbf{z}}) \le S_{\max}$ by Section 6.2. The route, squared-norm bound, response coordinate map, challenge family, and proof shape are fixed before the fold challenges.

5.6 Interaction and Special Cases

Figure 4 gives the order of the messages. The commitment handles and the fold description fix the source maps and the complete successor shape before any challenge. A different response certificate changes the collision bound that the incoming $\mathbf{A}$ map must support, and that map's frozen rank and geometry must already cover it. When provisioning a new commitment, we therefore resolve the certificate before choosing its rank, slices, and compression maps.

Transcript binding. Before any challenge, the preamble absorbs the setup-seed identity, the schedule digest, and the ordered commitment handles, points, projections, and claims; the digest binds every shape, coefficient order, challenge family and sampler, nonce budget, response route and integer encoding, compression path, and ring-check tag. Since the later messages follow Figure 4, no partial can be adapted to its scalar or fold challenge, and the successor witness cannot be adapted to α.

Ordinary openings and public batches. For an ordinary opening, take $G = 1$, $n_{\mathsf{clm},0} = 1$, $\pi_{0,0} = \mathrm{id}$, and $\phi_{0,0} = 1$. This leaves one response, no scalar-batching draw, and the single-claim witness order of (76). At the application root, the analyzed policy uses coefficient packing (127), and the ordered groups can refer to different commitments or to one commitment at several points. For the decoded vectors of a fixed extracted source, a discrepancy in any padded claim gives a nonzero polynomial in ζ_{batch} of degree at most $\Lambda - 1$; this is the public-batching argument of Lemma 10.33. At recursive edges, the same fold is instantiated with the source list of (30). The producer-side setup check and terminal binding are given in Sections 8.2 and 9.2.

Public input. Ordered groups $\mathcal{O}_h = (\mathcal{A}_g)_{g\in[G]}$, Λ claims in all. Group g has a handle cm_g, a point $\mathbf{r}_g$, and claims $\widetilde{F^{\mathsf{pad}}_{g,c}}(\mathbf{r}_g) = \phi_{g,c} v_{g,c}$, $\phi_{g,c} \neq 0$ (51). The schedule digest fixes Fold (53), edge_h, and the successor shape.
Witness. Source blocks $\mathbf{s}_{g,c,i}$, $i \in [B_g]$, with $\mathbf{t}_{g,c,i} = \mathbf{A}_g \mathbf{s}_{g,c,i}$, and the digits of each cm_g (Section 4.5).

---

0. Tensor reduction evaluation trace with $k > 1$ only

$\mathcal{P} \to \mathcal{V}$: column partials $S_{g,c,y}$; $\mathcal{V}$ checks $\phi_{g,c} v_{g,c} = \sum_y \widetilde{\mathsf{eq}}(y, \mathbf{r}_{g,\mathsf{head}}) S_{g,c,y}$.

$\mathcal{V} \to \mathcal{P}$: $\eta \leftarrow E^\kappa$ and $\zeta^{\mathsf{tensor}}_{g,c}$, the first equal to 1. One sum-check (54) ($m_{\max}$ rounds, degree 2) ends at ρ; $\rho_g := \rho_{[m_g]}$.

$\mathcal{P} \to \mathcal{V}$: $\tilde{v}_{g,c} = \theta_g \bar{f}_{g,c}(\rho_g)$. $\mathcal{V}$ checks $\theta_g \neq 0$ and the final value $\sum_{g,c} \zeta^{\mathsf{tensor}}_{g,c} \tilde{v}_{g,c}$.

---

1. Bind the partial opening evaluations

$\mathcal{P} \to \mathcal{V}$: $\mathbf{v}_{\mathsf{pub}}$, the image $\mathbf{v} = \mathbf{D}\hat{\mathbf{e}}$ (70) of the digits of all partials $e_{g,c,i}$ from (67) or (66), compressed by $\mathbf{H}_1, \mathbf{H}_2$ if $L_H = 2$.

---

2. Combine the scalar claims and fold each group

$\mathcal{V} \to \mathcal{P}$: weights $\vartheta_{g,c}$ (Section 5.4), giving the target v_{root} (72) or $\bar{v}_{\mathsf{root}} = \sum_{g,c} \vartheta_{g,c} \tilde{v}_{g,c}$.

$\mathcal{P} \to \mathcal{V}$: a nonce n from a set of size $N_{\mathsf{try},h}$. $\mathcal{V} \to \mathcal{P}$: challenges $c_{g,c,i}$, one query per coordinate under the round root and n; $c^A_{g,c,i} = \iota_g(c_{g,c,i})$ (packing) or $c_{g,c,i}$ (trace).

$\mathcal{P}$ keeps n only if every sampler succeeds and every $\mathbf{z}_g = \sum_{c,i} c^A_{g,c,i} \mathbf{s}_{g,c,i}$ (73) meets its certificate. $\mathcal{V}$ rejects a nonce outside the set or a failed sampler.

---

3. Commit to the successor witness

$\mathcal{P} \to \mathcal{V}$: $C_{W,h+1}$, the payload of $\mathbf{w}^{(h+1)}$ (76) under the successor shape (Figure 3). At the final nonterminal fold, instead $\mathbf{t}_i = \mathbf{A}_{\mathsf{term}} \mathbf{s}_i$ (Section 8.2).

---

4. Ring reduction, range, norm, and relation checks Figure 6

$\mathcal{V} \to \mathcal{P}$: $\alpha \leftarrow E$, $\tau_0 \leftarrow E^{\mu'}$, $\tau_1 \leftarrow E^{s_{\mathsf{rel}}}$. Write $\mathbf{w} := \mathbf{w}^{(h+1)}$ and $\mathbf{s}(x) := \mathbf{w}(x)(\mathbf{w}(x) + 1)$.

Stage 1. Product-tree sum-checks prove $\sum_x \widetilde{\mathsf{eq}}(\tau_0, x) Q_{\mathsf{sq}}(\mathbf{s}(x)) = 0$ (80) and leave $s_{\mathsf{claim}} = \tilde{\mathbf{s}}(\mathbf{r}_{\mathsf{virt}})$. On a Euclidean route, before the last of them, $\mathcal{P} \to \mathcal{V}$: $\mathcal{E}_{\mathsf{resp}}$ (83) and, if digit-expanded, the claims (86); $\mathcal{V}$ checks $\mathcal{E}_{\mathsf{resp}} \in [0, S_{\max}]$ and (88). The last sum-check then proves (90) under $\zeta_{\mathsf{pair}}, \zeta_{\mathsf{norm}} \leftarrow E$.

Stage 2. $\mathcal{V} \to \mathcal{P}$: η_{norm} (Euclidean route), ζ_{bin} (with compression), then γ. One sum-check (μ' rounds, degree 3) proves (125), merged with (93), and ends at $\mathbf{r}_2$. $\mathcal{P} \to \mathcal{V}$: $v_W = \widetilde{\mathbf{w}}^{(h+1)}(\mathbf{r}_2)$, and σ_S if offloaded. $\mathcal{V}$ checks the last round with $m_{\tau_1}(\mathbf{r}_2) = m_{\mathsf{local}}(\mathbf{r}_2) + \sigma_S$ (132), computing σ_S by (126) unless offloaded.

---

5. Pass the remaining claims to the next level

Direct: $\mathcal{O}_{h+1} = (\mathcal{W}_{h+1})$ with $\mathcal{W}_{h+1} = (C_{W,h+1}, \mathbf{r}_2, v_W)$.

Offloaded: Stage 3 sum-check (134) (ν_S rounds, degree 2) ends at ρ_S; $\mathcal{P} \to \mathcal{V}$: s_{ρ_S}; $\mathcal{V}$ checks (135). Then $\mathcal{O}_{h+1} = (\mathcal{S}_h, \mathcal{W}_{h+1})$ with $\mathcal{S}_h = (C_{S,h}, \rho_S, s_{\rho_S})$ (136).

Final: $\mathsf{TerminalInnerState}$: $(\mathbf{t}_i)_i$ and $\widetilde{\mathbf{w}}^{(h+1)}(\mathbf{r}_2) = v_W$; no setup claim.

Accept the fold iff every check passes.

Fig. 4: One Akita fold at a nonterminal level h, with outgoing claim $\widetilde{\mathbf{w}}^{(h+1)}(\mathbf{r}^{(h+1)}) = v^{(h+1)}$ for $(\mathbf{r}^{(h+1)}, v^{(h+1)}) := (\mathbf{r}_2, v_W)$.

6 Norm Checks for the Recursive Witness

The binding argument assumes that the next recursive witness is small in a publicly specified sense, and we certify that premise before the next fold. Every nonterminal fold proves a digit-range statement for the complete committed witness vector. A schedule may additionally select a Euclidean statement for the reconstructed response when the smaller collision bound justifies its cost. The algebraic equations of the fold are checked separately in Section 7.

6.1 Digit Range Check

At level $h+1$, the next commitment binds the witness vector $\mathbf{w} := \mathbf{w}^{(h+1)}$, indexed by $\{0,1\}^{\mu'}$. We write $\widetilde{\mathbf{w}} := \widetilde{\mathbf{w}}^{(h+1)}$ for its multilinear extension. The range check first certifies that every Boolean evaluation $\mathbf{w}(x)$, for $x \in \{0,1\}^{\mu'}$, lies in the common balanced alphabet

$$\mathcal{A}_{b^\star} := \{-b^\star/2, \ldots, b^\star/2 - 1\},$$

where $b^\star$ is the level's common range parameter, chosen by the schedule from $\{4, 8, 16, 32, 64\}$. A level's witness segments may use distinct positional bases for recomposition. The schedule separately records those bases and a single common range parameter $b^\star$ whose alphabet contains every balanced digit alphabet used by the witness entries. The range proof certifies this common bound; it does not identify the positional base of any segment. All compression digit spans in the witness vector are restricted to the smaller negative-binary alphabet $\{-1, 0\}$. These positions remain in the common range proof and are later narrowed by the fused binariness check in Remark 6.2.

The common range statement requires $Q_{b^\star}(\mathbf{w}(x)) = 0$ for every $x \in \{0,1\}^{\mu'}$, where

$$Q_{b^\star}(w) := \prod_{a \in \mathcal{A}_{b^\star}} (w - a).$$

Halving the predicate degree. A generic approach would evaluate the $b^\star$ linear factors of $Q_{b^\star}$ with a binary product tree, using $b^\star - 1$ multiplication gates per witness entry before proving that the output is zero at every Boolean point. We first reduce this product circuit by pairing the factors

$$(w - j)(w + j + 1) = w(w + 1) - j(j + 1),$$

for every $j = 0, \ldots, b^\star/2 - 1$. This rewrites the degree-$b^\star$ predicate in w as a degree-$b^\star/2$ predicate in the mapped value $s = w(w + 1)$. The intuition is that each pair of alphabet elements j and $-(j + 1)$ has the same image $j(j + 1)$ under this map. Thus, the product to be checked has only $b^\star/2$ factors rather than $b^\star$ factors.

For every Boolean point x, define the derived value

$$\mathbf{s}(x) := \mathbf{w}(x)\big(\mathbf{w}(x) + 1\big)$$

and the set of involution representatives $\mathcal{U}_{b^\star} := \{0, \ldots, b^\star/2 - 1\}$. If $\mathbf{w}(x)$ lies in the balanced alphabet $\mathcal{A}_{b^\star}$, then the corresponding derived value $\mathbf{s}(x)$ lies in the smaller set

$$\{j(j + 1) : j \in \mathcal{U}_{b^\star}\},$$

which has size $b^\star/2$. Hence, the vanishing predicate for these derived values is $Q_{\mathsf{sq}}(\mathbf{s}(x)) = 0$ for every $x \in \{0,1\}^{\mu'}$, where

$$Q_{\mathsf{sq}}(s) := \prod_{j \in \mathcal{U}_{b^\star}} \big(s - j(j + 1)\big). \tag{79}$$

The relation between the two predicate polynomials is $Q_{b^\star}(w) = Q_{\mathsf{sq}}\big(w(w + 1)\big)$.

The verifier samples a uniform point $\tau_0 \in E^{\mu'}$, which we call the *range point*. We certify the range through the degree-$(b^\star/2)$ identity

$$\sum_{x \in \{0,1\}^{\mu'}} \widetilde{\mathsf{eq}}(\tau_0, x) Q_{\mathsf{sq}}\big(\mathbf{s}(x)\big) = 0, \tag{80}$$

which combines the $2^{\mu'}$ vanishing conditions with the equality weights of τ_0. This identity can be proven by sum-checks, which reduce the range statement to the claim

$$s_{\mathsf{claim}} = \widetilde{\mathbf{s}}(\mathbf{r}_{\mathsf{virt}}).$$

We use $\mathbf{s}$ only as notation for these derived Boolean evaluations: we neither commit to them nor append them to the recursive witness. Their multilinear extension is

$$\widetilde{\mathbf{s}}(r) = \sum_{x \in \{0,1\}^{\mu'}} \widetilde{\mathsf{eq}}(r, x)\mathbf{w}(x)(\mathbf{w}(x) + 1).$$

We multiply at each Boolean index and then interpolate. Multiplying the interpolated values instead, as in $\widetilde{\mathbf{w}}(r)(\widetilde{\mathbf{w}}(r) + 1)$, generally gives a different value away from the Boolean hypercube. The ring-relation check later binds the claim on $\widetilde{\mathbf{s}}$ to the original committed witness $\mathbf{w}$ (Section 7), reducing both to a claim on $\widetilde{\mathbf{w}}$.

The product tree. For $b^\star \leq 8$, the polynomial Q_{sq} has degree at most four, so a single sum-check proves $Q_{\mathsf{sq}}(\mathbf{s}(x)) = 0$ for all $x \in \{0,1\}^{\mu'}$. This reduces the range claim directly to a claim about the multilinear polynomial $\widetilde{\mathbf{s}}$. Without norm fusion, this sum-check uses the equality-factored format of Section 3.5.

When $b^\star > 8$, we organize the product defining $Q_{\mathsf{sq}}(\mathbf{s}(x))$ as a tree of derived Boolean evaluations. At a Boolean point x, the leaves are the $b^\star/2$ factors $\mathbf{s}(x) - j(j+1)$ for $j \in \mathcal{U}_{b^\star}$, and each internal node value is the product of its child values at that same point. The root therefore evaluates to $Q_{\mathsf{sq}}(\mathbf{s}(x))$. Each node's multilinear polynomial interpolates these Boolean values; the pointwise product identities do not assert identities between these multilinear polynomials away from the Boolean hypercube. We choose the tree so that each product has degree either two or four: the root may have two children, and all remaining internal nodes have four children. Let d_i denote the product degree at level i, and let n_i denote the number of nodes at that level. We index the root by $i = 0$ and the leaves by the final level $i = S_{\mathsf{tree}}$. The resulting shapes are

$b^\star$	$(d_0, \ldots, d_{S_{\mathsf{tree}}})$	$(n_0, \ldots, n_{S_{\mathsf{tree}}})$
4	$(2, 0)$	$(1, 2)$
8	$(4, 0)$	$(1, 4)$
16	$(2, 4, 0)$	$(1, 2, 8)$
32	$(4, 4, 0)$	$(1, 4, 16)$
64	$(2, 4, 4, 0)$	$(1, 2, 8, 32)$.

Here $d_{S_{\mathsf{tree}}} = 0$ because the final level consists of leaves.

The protocol proves the root claim by moving down the tree. At product level i, for $0 \leq i < S_{\mathsf{tree}}$, each node at that level gives one relation saying that the parent value at every Boolean point is the product of its child values. These relations all have the same degree d_i, so they can be batched into one equality-factored sum-check without increasing the degree, provided no norm term is fused into that stage. The output of this batched sum-check is a collection of evaluation claims on the multilinear polynomials interpolating the child values at level $i + 1$.

If level $i + 1$ is not the leaf level, the same reduction is repeated at the next product level. At the leaf level, the values $\mathbf{s}(x) - j(j+1)$ are all affine functions of the same derived Boolean values $\mathbf{s}$. Therefore the final leaf claims collapse to a single claim about $\widetilde{\mathbf{s}}$ at the final random point,

$$s_{\mathsf{claim}} = \widetilde{\mathbf{s}}(\mathbf{r}_{\mathsf{virt}}).$$

This is the only claim that leaves the range pipeline.

As a result, there are S_{tree} sequential range sum-checks, one for each product level $i = 0, \ldots, S_{\mathsf{tree}} - 1$. Without norm fusion, all use the equality-factored format. On a Euclidean route, the last stage (the only stage when $b^\star \leq 8$) has the same rounds but uses the standard compressed message format. The number of intermediate child evaluations passed between product levels is controlled by $\sum_{i=1}^{S_{\mathsf{tree}}-1} n_i$. For example, when $b^\star/2 = 32$, our tree has intermediate node counts $2 + 8$. A fully binary tree over the same 32 factors would have intermediate node counts $2 + 4 + 8 + 16$.

Verifier offloading (Section 9.1) does not change this range tree.

Remark 6.1 (Digit range choice). In practice, we use the negative-biased balanced range $\{-b^\star/2, \ldots, b^\star/2 - 1\}$ for a power-of-two base $b^\star \in \{4, 8, 16, 32, 64\}$. The balanced range minimizes the digit norm for a fixed alphabet size. When $b^\star$ is even, its negative bias also enables the degree-halving identity in (79), since the two roots j and $-(j+1)$ have the same image under $w \mapsto w(w+1)$. The power-of-two choice makes digit decomposition a sequence of shifts and masks, improving practical efficiency.

Remark 6.2 (Compression-digit binariness via the fused sum-check). Every compression digit is restricted to $\{-1, 0\}$ (Section 4.5) and therefore requires an additional binariness proof on these positions. Its vanishing polynomial $w(w+1)$ is the same quadratic used for the derived values $\mathbf{s}(x) = \mathbf{w}(x)(\mathbf{w}(x)+1)$, so the proof needs neither a separate sum-check nor a higher individual degree.

After the range pipeline fixes $s_{\mathsf{claim}} = \widetilde{\mathbf{s}}(\mathbf{r}_{\mathsf{virt}})$, the verifier samples a coefficient ζ_{bin} for the binariness claim and then a separate coefficient γ for the claim s_{claim}. The fused sum-check (125) combines both claims with the ring relation. Its public binariness weight uses $\widetilde{\mathsf{eq}}_{\mathcal{I}_{\mathsf{bin}}}(\mathbf{r}_{\mathsf{virt}}, X)$, the MLE in X of the Boolean function $x \mapsto \mathbf{1}_{\mathcal{I}_{\mathsf{bin}}}(x)\widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{virt}}, x)$. This restricted equality is not the product of the separate MLEs of equality and the support indicator.

The set $\mathcal{I}_{\mathsf{bin}}$ consists of the witness positions of the compression digit vectors $\hat{\mathbf{u}}_j$ and $\hat{\mathbf{v}}_j$ of (47). The public witness layout fixes these positions as a union of contiguous spans, so the verifier evaluates the restricted equality weight without enumerating the padded witness; Appendix C.2 gives the exact verifier calculation.

6.2 Certifying the ℓ_2 Response Norm

The digit range check bounds every reconstructed response coefficient but allows all of them to be large at once. On the Euclidean route, we also certify the squared norm of the reconstructed response, which can reduce the Module-SIS parameters of the inner commitment. A digit in position a contributes b^a times its value to its coefficient, where b is the response base, and these contributions are added before squaring. The digit range check remains in force for every witness segment.

The response norm statement. Let m_A be the input width of $\mathbf{A}$ and d_A its ring dimension. The reconstructed response $\mathbf{z}$ has $N_A := m_A d_A$ base-field coordinates, indexed by $u \in [N_A]$; these are the coefficients of $\mathbf{z}$, not its stored digits $\hat{\mathbf{z}}$. The digits of each coefficient are located in the committed witness by the injective map

$$\pi : [N_A] \times [\delta_f] \longrightarrow [|\mathbf{w}^{(h+1)}|]$$

whose image is exactly the scheduled $\hat{\mathbf{z}}$ digit cells. For the response base b, define the signed integer response coordinate

$$z_u^{\mathbb{Z}} := \sum_{a=0}^{\delta_f - 1} b^a \mathbf{w}^{(h+1)}\big(\pi(u, a)\big). \tag{81}$$

Reducing $z_u^{\mathbb{Z}}$ modulo q gives coordinate u of the ring vector $\mathbf{z} = \mathbf{G}_{b,m_A}\hat{\mathbf{z}}$. Centered reduction cannot increase its absolute value. Hence the exact integer quantity

$$\mathcal{E}_{\mathsf{int}}(\hat{\mathbf{z}}) := \sum_{u=0}^{N_A - 1} \big(z_u^{\mathbb{Z}}\big)^2 \quad \text{satisfies} \quad \|\mathbf{z}\|_{2,\mathrm{coef}}^2 \le \mathcal{E}_{\mathsf{int}}(\hat{\mathbf{z}}). \tag{82}$$

The Euclidean route requires the exact statement

$$\mathcal{E}_{\mathsf{int}}(\hat{\mathbf{z}}) = \mathcal{E}_{\mathsf{resp}} \qquad \text{and} \qquad \mathcal{E}_{\mathsf{resp}} \le S_{\max}, \tag{83}$$

where $S_{\max}$ is fixed by the schedule. The transcript carries $\mathcal{E}_{\mathsf{resp}}$ in the canonical fixed-width unsigned encoding determined by $S_{\max}$. The verifier rejects any noncanonical encoding and any decoded value outside $[0, S_{\max}]$. The index set in (82) contains every coefficient used by $\mathbf{A}$ exactly once; the norm is not taken per ring row or per witness part. The map π is defined after extension-field packing, so no further packing factor is applied to this norm.

For the sum-checks below, extend the reconstructed response to the Boolean domain $\{0,1\}^{\mu'}$ by placing the N_A coordinates in their scheduled order and setting all remaining positions to zero. We write $z_{\mathsf{int}}(x)$ for the resulting Boolean evaluation vector and $\widetilde{z_{\mathsf{int}}}$ for its multilinear extension. The schedule fixes both this order and the map π.

Direct norm check. A sum-check proves an equality in $\mathbb{F}_q$. It determines the integer squared norm when both sides lie in $[0, q)$, which the schedule checks in advance from the certified digit ranges. For each response plane a, let $B_{\mathsf{dig},a}$ be the largest absolute integer admitted by its scheduled digit alphabet, and define

$$U_{\mathsf{dir}} := N_A \left(\sum_{a=0}^{\delta_f - 1} b^a B_{\mathsf{dig},a} \right)^2 . \tag{84}$$

The direct route is admissible only when $\max\{U_{\mathsf{dir}}, S_{\max}\} < q$. The prover sends the canonical encoding of $\mathcal{E}_{\mathsf{resp}}$ and proves the field identity

$$\sum_{x \in \{0,1\}^{\mu'}} z_{\mathsf{int}}(x)^2 = \mathcal{E}_{\mathsf{resp}}. \tag{85}$$

Lemma 6.3 (Direct norm check). *Assume the range check accepts, the map π is injective with the scheduled image, and $\max\{U_{\mathsf{dir}}, S_{\max}\} < q$. Then (85), together with the verifier's canonical decoding and comparison against $S_{\max}$, implies*

$$\mathcal{E}_{\mathsf{int}}(\hat{\mathbf{z}}) = \mathcal{E}_{\mathsf{resp}} \le S_{\max}$$

over the integers.

Digit-expanded norm reconstruction. When the squared norm can exceed q, we expand it into inner products between digit planes, where plane a collects digit a of every response coefficient. Each inner product is taken over a short segment, so its integer value lies in $(-q/2, q/2)$ and is determined by its residue, and the verifier reconstructs the norm over the integers. Write the Boolean digit vectors as $z_a(x) := \mathbf{w}^{(h+1)}(\pi(x, a))$ for $a \in [\delta_f]$, with zero padding outside $[N_A]$. Partition $[N_A]$ into public consecutive segments I_t. For every segment and every $0 \le a \le a' < \delta_f$, the prover supplies the claim

$$p_{t,a,a'} := \sum_{u \in I_t} z_a(u) z_{a'}(u) \in \mathbb{F}_q. \tag{86}$$

The schedule must choose the partition so that, for every (t, a, a'),

$$|I_t| B_{\mathsf{dig},a} B_{\mathsf{dig},a'} < q/2. \tag{87}$$

The verifier takes the unique centered integer lift $\bar{p}_{t,a,a'} \in (-q/2, q/2)$ and reconstructs

$$\mathcal{E}_{\mathsf{resp}} = \sum_t \left(\sum_{a=0}^{\delta_f - 1} b^{2a} \bar{p}_{t,a,a} + 2 \sum_{0 \le a < a' < \delta_f} b^{a+a'} \bar{p}_{t,a,a'} \right). \tag{88}$$

The verifier compares the reconstruction with the decoded integer $\mathcal{E}_{\mathsf{resp}}$ and rejects if they differ, if the reconstruction is negative, or if it exceeds $S_{\max}$.

Lemma 6.4 (Unique integer lifting of digit-inner-product claims). *Assume the response digit ranges and (87). If (86) holds in $\mathbb{F}_q$, then $\bar{p}_{t,a,a'}$ is the integer inner product $\sum_{u \in I_t} z_a(u) z_{a'}(u)$.*

Proposition 6.5 (Integer norm reconstruction from digit inner products). *If every digit-inner-product claim has its true centered lift, then the right-hand side of (88) equals $\mathcal{E}_{\mathsf{int}}(\hat{\mathbf{z}})$ over $\mathbb{Z}$. Consequently, comparison with the canonical value $\mathcal{E}_{\mathsf{resp}} \le S_{\max}$ certifies (83).*

The proofs of Lemmas 6.3 and 6.4 and Proposition 6.5 are in Appendix D.1.

Fusion with the final range-tree sum-check. The norm proof shares the last sum-check of the range tree. The prover sends the norm claims after the earlier range-tree stages, immediately before this last sum-check, so they may depend on α, τ_0, τ_1, and the earlier range challenges. Each batching challenge below is drawn after the claims it combines: the digit claims precede ζ_{pair}, both merged claims precede ζ_{norm}, and the response evaluations precede η_{norm}. Let $P_{\mathsf{rng}}(x)$ denote the summand in that final range leaf, including its equality factor, and let C_{rng} be its current claim. In the direct case, set

$$P_{\mathsf{norm}}(x) := z_{\mathsf{int}}(x)^2, \qquad C_{\mathsf{norm}} := \mathcal{E}_{\mathsf{resp}}.$$

In the digit-expanded case, the verifier first samples $\zeta_{\mathsf{pair}} \in E$ after all digit-inner-product claims are fixed. Let $\mathsf{idx}(t, a, a')$ be the position of the triple (t, a, a') in a public canonical ordering. The number of triples is

$$N_{\mathsf{pair}} := |\{(t, a, a') : 0 \le a \le a' < \delta_f\}| = |\{I_t\}| \, \frac{\delta_f(\delta_f + 1)}{2}. \tag{89}$$

Using this ordering, the verifier sets

$$P_{\mathsf{norm}}(x) := \sum_{t, a \le a'} \zeta_{\mathsf{pair}}^{\mathsf{idx}(t,a,a')} \mathbf{1}_{I_t}(x) z_a(x) z_{a'}(x),$$
$$C_{\mathsf{norm}} := \sum_{t, a \le a'} \zeta_{\mathsf{pair}}^{\mathsf{idx}(t,a,a')} p_{t,a,a'}.$$

After these claims are fixed, the verifier samples a fresh $\zeta_{\mathsf{norm}} \in E$ and runs one sum-check for

$$\sum_{x \in \{0,1\}^{\mu'}} (P_{\mathsf{rng}}(x) + \zeta_{\mathsf{norm}} P_{\mathsf{norm}}(x)) = C_{\mathsf{rng}} + \zeta_{\mathsf{norm}} C_{\mathsf{norm}}. \tag{90}$$

Fusion adds no sum-check rounds. The direct norm term has degree two and each digit-pair term degree three, at most the degree $d_i + 1$ of the last range summand with its equality factor ($d_i \in \{2, 4\}$). Since the norm term does not carry that equality factor, the fused stage uses the standard compressed format, which sends $d_i + 1$ field elements per round, one more than the equality-factored range message (Table 4).

Binding the norm values to the committed witness. At the final range-tree sum-check point $\mathbf{r}_{\mathsf{virt}}$, the direct proof leaves the evaluation $\widetilde{z_{\mathsf{int}}}(\mathbf{r}_{\mathsf{virt}})$. The digit-expanded proof leaves the digit evaluations $\widetilde{z_a}(\mathbf{r}_{\mathsf{virt}})$ for $a \in [\delta_f]$. These values must be tied to the $\hat{\mathbf{z}}$ cells inside the committed witness. After these evaluations are fixed, the verifier samples a fresh batching challenge $\eta_{\mathsf{norm}} \in E$. The required randomized identities are

$$\eta_{\mathsf{norm}} \widetilde{z_{\mathsf{int}}}(\mathbf{r}_{\mathsf{virt}}) = \eta_{\mathsf{norm}} \sum_{u \in [N_A]} \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{virt}}, u) \sum_{a=0}^{\delta_f - 1} b^a \mathbf{w}^{(h+1)}\big(\pi(u, a)\big), \tag{91}$$

$$\sum_{a=0}^{\delta_f - 1} \eta_{\mathsf{norm}}^{a+1} \widetilde{z_a}(\mathbf{r}_{\mathsf{virt}}) = \sum_{u \in [N_A]} \sum_{a=0}^{\delta_f - 1} \eta_{\mathsf{norm}}^{a+1} \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{virt}}, u) \mathbf{w}^{(h+1)}\big(\pi(u, a)\big) \tag{92}$$

for the direct and digit-expanded routes, respectively. Both right-hand sides are public linear functions of the same witness used by the relation check. Let C_{base} and P_{base} denote the input claim and summand of the ordinary relation and range-binding sum-check (Section 7.4). Let C_{bind} be the left-hand side of (91) in the direct case or of (92) in the digit-expanded case, and let P_{bind} be the corresponding right-hand summand after applying the public response-digit map. The combined sum-check proves

$$\sum_{x} (P_{\mathsf{base}}(x) + P_{\mathsf{bind}}(x)) = C_{\mathsf{base}} + C_{\mathsf{bind}}. \tag{93}$$

The ordinary relation is independent of η_{norm}, while every norm-check residual has positive degree in η_{norm}. Thus a false norm-check identity cannot cancel a false ordinary relation identically in the batching challenge. The combined sum-check reduces the range, norm, and relation statements to the one evaluation claim on $\widetilde{\mathbf{w}}^{(h+1)}$ already carried by the recursive protocol.

Terminal response and security interface. The evaluation-trace terminal has no next witness, so it checks the revealed response against its scheduled bound directly (Section 8.2). The soundness error of the two norm protocols is accounted for in Lemma 10.34, and Corollary 10.35 turns the certified premise (83) into the route-specific $\mathbf{A}$ collision radius. Honest-prover bounds used to select $S_{\max}$ are stated in Section 10.2 and are not part of this soundness argument.

7 Checking the Ring Relations

The norm checks of Section 6 bound the recursive witness but do not show that it comes from a valid fold. This section checks the fold, commitment, and evaluation equations on the same witness digits. The evaluations give scalar rows over E and the other equations give ring rows (Figure 5); after reducing the ring rows to field rows, one sum-check combines all of them with the range-check output.

7.1 Opening-Method Evaluation Rows

We first write each claimed evaluation as a scalar row on the opening digits, for one claim with the group and claim indices suppressed.

Direct coefficient packing. Replacing each $e_{i,t,j}$ in the scalar opening (59) by its digit expansion $\sum_\ell b_{\mathsf{op}}^\ell \hat{e}_{i,\ell,t,j}$, where $b_{\mathsf{op}} := b_1$ is the opening-digit base, gives

$$\sum_{\substack{i\in[B],\ \ell\in[\delta_{\mathsf{open}}]\\ t\in[k],\ j\in[d_{\mathsf{car}}]}} \underbrace{\widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{blk}}, i)\, b_{\mathsf{op}}^\ell\, \beta_t\, \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{tail}}, j)}_{\omega_{\mathsf{dir}}(i,\ell,t,j)}\, \hat{e}_{i,\ell,t,j} = v, \tag{94}$$

with $\hat{e}_{i,\ell,t,j} \in K$. This is one E-valued row, checked directly over E with neither a quotient nor an α-weight. The packing-consistency equation (63) remains a separate ring-valued row. Appendix C.2 evaluates the weight of this row at the final sum-check point without constructing it.

Evaluation-trace row. At a nonterminal evaluation-trace fold, the tensor-reduction prefix first fixes a reduced claim $\bar{f}(\rho) = \bar{v}$, and the partials $e_i \in R_{q,d_A}$ of (66) have already contracted ρ_{pos}. Of the remaining coordinates in (65), ρ_{blk} gives block i the weight

$$\chi_{\mathsf{blk}}(i) := \widetilde{\mathsf{eq}}(\rho_{\mathsf{blk}}, i), \qquad i \in [B], \tag{95}$$

and ρ_{pack} evaluates the coordinates packed into one ring element. Packing its equality weights with the trace-packing map $\psi : E^{d_A/k} \to R_{q,d_A}$ gives

$$\check{\chi}_{\mathsf{pack}} := \psi\Big(\big(\widetilde{\mathsf{eq}}(\rho_{\mathsf{pack}}, u)\big)_{u\in[d_A/k]}\Big), \tag{96}$$

and the public linear functional

$$\mathcal{T}_{\rho_{\mathsf{pack}}}(Z) := \frac{k}{d_A}\,\mathrm{Tr}_H(Z\,\sigma_{-1}(\check{\chi}_{\mathsf{pack}}))\,. \tag{97}$$

The trace identity of Hachi [80, Theorem 2] turns the packed opening into the field row

$$\sum_{i\in[B]} \chi_{\mathsf{blk}}(i)\,\mathcal{T}_{\rho_{\mathsf{pack}}}(e_i) = \bar{v}. \tag{98}$$

By linearity of the trace, the weight on opening digit (i,ℓ,ν), for full-ring coefficient $\nu \in [d_A]$ and digit ℓ, is

$$\omega_{\mathsf{Tr}}(i,\ell,\nu) := \chi_{\mathsf{blk}}(i) b_{\mathsf{op}}^\ell \mathcal{T}_{\rho_{\mathsf{pack}}}(X^\nu). \tag{99}$$

When the final tensor-reduction check is fused with this row, both sides of (98) are multiplied by the public accumulator value $A_\eta(\rho)$ of Section 3.7, which the tensor-reduction verifier has checked to be nonzero, so no inversion is needed. In the shared reduction this factor is θ_g, which also contains the equality factor on any extra common variables; the identity reduction uses $\theta_g = 1$ (Section 5.1).

One scalar row for the batch. Restoring the group and claim indices, the packing rows are weighted by $\vartheta_{g,c}$ and compared with the target of (72):

$$\sum_{g,c} \vartheta_{g,c} \sum_{i\in[B_g]} \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{blk},g,c}, i) \sum_{u\in[d_{\mathsf{car},g}]} \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{tail},g,c}, u) \sum_{t\in[k]} \beta_t \sum_{\ell} b^{\ell}_{1,g} \hat{e}_{g,c,i,\ell,t,u} = v_{\mathsf{root}}. \tag{100}$$

Every summand uses its own point and challenge-subring dimension.

With evaluation trace, the corresponding target and scalar row are

$$\bar{v}_{\mathsf{root}} := \sum_{g\in[G]} \sum_{c\in[n_{\mathsf{clm},g}]} \vartheta_{g,c} \tilde{v}_{g,c}, \qquad \sum_{g\in[G]} \sum_{c\in[n_{\mathsf{clm},g}]} \vartheta_{g,c} \theta_g \boldsymbol{\omega}^{\top}_{\mathsf{Tr},g} \hat{\mathbf{e}}_{g,c} = \bar{v}_{\mathsf{root}}, \tag{101}$$

where $\boldsymbol{\omega}_{\mathsf{Tr},g}$ is derived from the group-local point ρ_g and evaluation-trace packing shape. Neither row imposes a common opening point. The $\mathbf{D}/\mathbf{H}$ payload binds $\hat{\mathbf{e}}$ before the fold challenges, and the selected scalar row joins the relation sum-check as an ordinary row; Section 10.3 accounts for its errors.

7.2 The Common Ring Relation

The remaining rows tie the partials and responses to the incoming commitments. Each group contributes its fold-evaluation and inner-consistency rows and its sliced outer path; only the opening path and scalar row span all groups. Each row uses the ring dimension and coefficient field of its own map, as fixed by the schedule, so no row is padded to the largest ring.

Recompose $\mathbf{z}_g$ by (74), recompose $\mathbf{t}_{g,c,i}$ from its opening digits, and recover each $e_{g,c,i}$ from its base-field coefficient planes. For a compressed path, its first digit vector also recomposes the image $\mathbf{u}_g$ or $\mathbf{v}$; for an uncompressed path, that image is the public payload. The equations in Figure 5 are therefore linear in the complete witness of (76) once the public challenges are fixed.

The scalar row and all ring rows form one field relation $\mathsf{M}(\alpha)\mathbf{w} = \mathbf{y}(\alpha)$ after the selected ring reduction. At the root we also write this matrix and target as $\mathsf{M}_{\mathsf{root}}$ and $\mathbf{y}_{\mathsf{root}}$. For $G = 1$ and one claim, the rows specialize to the local formulas derived in Sections 5.3 and 7.1; no extra batching coefficient or witness segment appears.

7.3 Reducing Ring Equations to Field Identities

An equality modulo $X^d + 1$ need not remain an equality after a field challenge is substituted for X, so the ring rows need a reduction before a field sum-check. We use two. Quotient lifting adds a private quotient polynomial, which the recursion must carry. Quotient-free checking places the reduction modulo $X^d + 1$ in the verifier's public weights instead. Both start from one normal form.

A Common Normal Form for the Ring Rows Each ring row multiplies public ring elements by polynomials whose coefficients sit, not necessarily consecutively, in the recursive witness, and compares the sum with a public target. Let $\mathsf{Rows}_{\mathsf{ring}}$ be the ordered set of active ring-valued rows in Figure 5. A row $r \in \mathsf{Rows}_{\mathsf{ring}}$ has its own native dimension d_r and coefficient field $K_r \subseteq E$. We write its ring as

$$R_r := K_r[X]/(X^{d_r} + 1).$$

The $\mathbf{A}, \mathbf{B}, \mathbf{D}, \mathbf{F}$, and $\mathbf{H}$ rows have $K_r = K = \mathbb{F}_q$; the carrier row of coefficient packing has $K_r = E$ and is represented by its $\mathbb{F}_q$-basis planes. The schedule fixes d_r, the row's position in the canonical row order, and every witness interval it reads. We use

$$d_{\max} := \max_{r\in\mathsf{Rows}_{\mathsf{ring}}} d_r \tag{107}$$

for the largest native dimension among these active rows.

Let $N_{\mathsf{w}} := |\mathbf{w}|$ and write $\mathbf{w} = (w_u)_{u\in[N_{\mathsf{w}}]}$ for the outgoing witness of (76), before its zero padding to length $2^{\mu'}$, with its base-field entries embedded canonically in K_r. A row reads several polynomials, indexed

The relation for one opening batch.
The following families are ordered as listed. Within a family, use increasing group, slice, and matrix-row indices. Write $\sum_{c,i}$ for $\sum_{c\in[n_{\mathsf{clm},g}]}\sum_{i\in[B_g]}$ in a group-local row.

1. Fold evaluation, one row per group. With coefficient packing, use the local contraction and challenge embedding:

$$\sum_{c,i} c_{g,c,i}(Y)e_{g,c,i}(Y) = \mathcal{L}_g(\mathbf{G}_{b_g,M_g}\mathbf{z}_g)(Y) \quad \text{in } E[Y]/(Y^{d_{\mathsf{car},g}}+1). \tag{102}$$

With evaluation trace, use the full-ring row instead:

$$\sum_{c,i} c_{g,c,i}(X)e_{g,c,i}(X) = \mathbf{a}_g^{\top}\mathbf{G}_{b_g,M_g}\mathbf{z}_g(X) \quad \text{in } R_{q,d_{A,g}}. \tag{103}$$

A carrier row has k base-field coordinate planes; its native modulus remains $Y^{d_{\mathsf{car},g}}+1$.

2. Inner consistency, $n_{A,g}$ rows per group.

$$\sum_{c,i} c^{A}_{g,c,i}\mathbf{t}_{g,c,i} = \mathbf{A}_g\mathbf{z}_g \quad \text{in } R_{q,d_{A,g}}^{n_{A,g}}. \tag{104}$$

Here $c^{A}_{g,c,i} = \iota_g(c_{g,c,i})$ for packing and $c^{A}_{g,c,i} = c_{g,c,i}$ for evaluation trace.

3. Outer consistency, one matrix equation per group and slice.

$$\mathbf{B}_g\mathbf{x}_{B,g,s} = \mathbf{u}_{g,s}, \qquad g\in[G], \quad s\in[S_{B,g}]. \tag{105}$$

The complete image $\mathbf{u}_g = \|_s\,\mathbf{u}_{g,s}$ is either the raw payload or the recomposition of $\hat{\mathbf{u}}_{g,1}$. Every slice reuses the same matrix $\mathbf{B}_g$.

4. Shared opening consistency, one matrix equation.

$$\mathbf{D}\,\mathsf{concat}(\hat{\mathbf{e}}_0,\dots,\hat{\mathbf{e}}_{G-1}) = \mathbf{v}. \tag{106}$$

The image is the raw payload or the recomposition of $\hat{\mathbf{v}}_1$. This map is unsliced and uses the exact summed width.

5. Compression chains, in increasing depth. Apply (47) to each $\mathbf{B}_g/(\mathbf{F}_{g,a})_a$ path and the shared $\mathbf{D}/(\mathbf{H}_a)_a$ path. Families 3 and 4 already supply the first row of each chain. At each remaining depth, put all group-local $\mathbf{F}$ rows before the shared $\mathbf{H}$ rows. The final rows have public targets $\mathbf{u}_{\mathsf{pub},g}$ and $\mathbf{v}_{\mathsf{pub}}$. At depth zero, there are no compression rows.

6. One scalar opening row. Use (100) for coefficient packing or (101) for evaluation trace. This is an E-linear row and has no quotient.

The quotient-lift route carries quotient digits for families 1–4 before the compression stages, then carries each stage's quotient digits just after its binary digits, using each row's native modulus and coefficient field. The quotient-free route carries none. Both use the witness order of (75) and (76).

Fig. 5: The canonical relation for one fold. Group-local rows preserve the incoming commitment shapes and points; the shared opening and scalar rows connect them to one successor witness.

by a finite set Occ_r, and one witness coordinate may contribute to several of them. For occurrence $c \in \mathsf{Occ}_r$, the address map specifies which witness coordinate supplies its coefficient of X^j:

$$\mathsf{addr}_{r,c} : [d_r] \longrightarrow [N_{\mathsf{w}}] \tag{108}$$

including the scheduled subcolumn, digit, and compression-layer coordinates. Reading the witness at these addresses gives

$$W_{r,c}(X) := \sum_{j=0}^{d_r-1} w_{\mathsf{addr}_{r,c}(j)} X^j \in R_r. \tag{109}$$

All gadget powers, folding challenges, row signs, and public projection weights are placed in a public multiplier $A_{r,c}(X) \in R_r$. For the carrier row, these multipliers include the fixed basis elements β_t that combine its base-field coordinate planes. Thus every ring-valued row has the form

$$Z_r(X) := \sum_{c \in \mathsf{Occ}_r} A_{r,c}(X) \circledast_{d_r} W_{r,c}(X) - Y_r(X) = 0 \quad \text{in } R_r, \tag{110}$$

where $Y_r \in R_r$ is public and $\circledast_d$ denotes negacyclic multiplication modulo $X^d + 1$. In Figure 5, group g's fold-evaluation row has native dimension $d_{\mathsf{car},g}$ with coefficient packing and $d_{A,g}$ with evaluation trace; its inner-consistency rows use $d_{A,g}$, its sliced outer rows $d_{B,g}$, the opening-consistency rows d_D, and each compression row the dimension of its own $\mathbf{F}_j$ or $\mathbf{H}_j$ map. In a matrix row, the setup entries supply coefficients of $A_{r,c}$ and the opposite side has gadget-recomposition multipliers. The scalar opening row is already E-linear, is not in $\mathsf{Rows}_{\mathsf{ring}}$, and joins the field relation after the ring rows are reduced.

Quotient Lifting As in Lemma 3.4, take degree-$<$ d_r representatives in (110). The prover supplies a quotient $Q_r \in K_r[X]$ of degree less than d_r for the identity

$$\sum_{c \in \mathsf{Occ}_r} A_{r,c}(X) W_{r,c}(X) - Y_r(X) = (X^{d_r} + 1) Q_r(X), \tag{111}$$

in $K_r[X]$; a valid row admits one, since each product on the left has degree at most $2d_r - 2$. The next commitment binds the folded witness and every quotient before the verifier samples $\alpha \leftarrow E$, and evaluating at $X = \alpha$ gives

$$\sum_{c \in \mathsf{Occ}_r} \sum_{j=0}^{d_r-1} A_{r,c}(\alpha) \alpha^j w_{\mathsf{addr}_{r,c}(j)} - Y_r(\alpha) - (\alpha^{d_r} + 1) Q_r(\alpha) = 0. \tag{112}$$

The witness weight factors into $A_{r,c}(\alpha)$ and the universal coefficient weight α^j, at the price of the private Q_r, whose digits join the next recursive witness.

We specialize to the fold rows of one group, with i ranging over its claim–block pairs (Section 5.1). With coefficient packing, the fold-evaluation row is over the carrier ring of degree d_{car} with coefficient field E, and its quotient is written in the basis $\boldsymbol{\beta}$ as

$$Q_{\mathsf{car}}(Y) = \sum_{t \in [k]} \beta_t \sum_{j \in [d_{\mathsf{car}}]} q_{t,j} Y^j.$$

The evaluated row is

$$\sum_i c_i(\alpha) e_i(\alpha) - \mathcal{L}(\mathbf{G}_{b,M} \mathbf{z})(\alpha) - (\alpha^{d_{\mathsf{car}}} + 1) Q_{\mathsf{car}}(\alpha) = 0. \tag{113}$$

The k planes $(q_{t,j})$ form one polynomial over E, so the modulus factor stays $\alpha^{d_{\mathsf{car}}} + 1$, not $\alpha^{kd_{\mathsf{car}}} + 1$. In the inner-consistency rows the challenge enters through $Y \mapsto X^{kh_{\mathsf{car}}}$, so an $\mathbf{A}$ row $r \in [n_A]$ becomes

$$[\mathbf{A}(\alpha)\mathbf{z}(\alpha)]_r - \sum_i c_i(\alpha^{kh_{\mathsf{car}}}) [\mathbf{t}_i]_r(\alpha) - (\alpha^{d_A} + 1) Q_{A,r}(\alpha) = 0, \tag{114}$$

where $\mathbf{A}(\alpha)$ and $\mathbf{z}(\alpha)$ evaluate the degree-$< d_A$ representatives entrywise. The same challenge is thus evaluated as $c_i(\alpha)$ in the carrier row and as $c_i(\alpha^{kh_{\mathsf{car}}})$ in the $\mathbf{A}$ rows; the two coincide only when $kh_{\mathsf{car}} = 1$. With evaluation trace, the challenges lie in the full ring, both fold rows use $c_i(\alpha)$, and the fold-evaluation row is

$$\sum_i c_i(\alpha)e_i(\alpha) - \mathbf{a}(\alpha)^\top \mathbf{G}_{b,M}\mathbf{z}(\alpha) - (\alpha^{d_A}+1)Q_{\mathsf{eval}}(\alpha) = 0, \tag{115}$$

and the $\mathbf{A}$ rows use $c_i(\alpha)$. With either method, the $\mathbf{B}, \mathbf{D}, \mathbf{F}$, and $\mathbf{H}$ families use the ordinary lift at their scheduled dimensions, and the scalar opening row joins the field relation unchanged.

Soundness for mixed ring dimensions. These row families have different ring dimensions but share the single challenge α. Fix ring dimensions $d_1, \ldots, d_J$, each a power of two, and write $R_q^{(d_b)} := \mathbb{Z}_q[X]/(X^{d_b}+1)$.

Let a relation consist of row blocks $b = 1, \ldots, J$, where block b asserts $\mathsf{M}^{(b)}\,\mathbf{w}^{(b)} = \mathbf{h}^{(b)}$ over $R_q^{(d_b)}$, and each $\mathbf{w}^{(b)}$ is the flat $\mathbb{Z}_q$-coefficient vector of one committed vector $\mathbf{w}$ read as a vector of d_b-coefficient ring elements (distinct blocks may read overlapping coordinates of $\mathbf{w}$ under distinct dimensions). Fix the matrices, witness, targets, and prover-supplied quotient vectors $\mathbf{r}^{(b)}$ of degree less than d_b before sampling the single challenge $\alpha \in E$. Define the per-block evaluated identity

$$\mathsf{M}^{(b)}(\alpha)\,\mathbf{w}^{(b)}(\alpha) = \mathbf{h}^{(b)}(\alpha) + (\alpha^{d_b}+1)\,\mathbf{r}^{(b)}(\alpha), \tag{116}$$

where, as in (26), the block-b witness evaluation is the partial evaluation of $\widetilde{\mathbf{w}}$ against the truncated weight $\widetilde{\alpha}^{(d_b)}$, the multilinear extension of $(1, \alpha, \ldots, \alpha^{d_b-1})$ on $\log d_b$ coefficient variables.

Lemma 7.1 (Mixed-dimension ring switch). *For the relations and fixed quotient vectors above:*

1. (Completeness.) *If every block relation holds over its ring, the honest quotients of Lemma 3.4 (computed blockwise, with modulus $X^{d_b}+1$) satisfy (116) for every α.*
2. (Soundness.) *If some block relation fails, then for any choice of quotients the identities (116) hold simultaneously for at most $2\,d_{\max}-1$ values of α, where $d_{\max} := \max_b d_b$; equivalently, the step is $2d_{\max}$-special sound in α and a cheating witness passes the check at a uniformly sampled $\alpha \leftarrow E$ with probability at most $(2\,d_{\max}-1)/|E|$.*
3. (One ladder.) *All truncated weights are prefixes of one tensor: $\widetilde{\alpha}^{(d_b)}(y) = \prod_{j=0}^{\log d_b - 1}\big(1 + y_j(\alpha^{2^j}-1)\big)$, so the verifier computes the single power ladder $\{\alpha^{2^j}\}_{j<\log d_{\max}}$ once and each block uses its length-$\log d_b$ prefix.*

The proof is in Appendix D.2. The quotient coefficients are appended to the committed vector before α is sampled, and block b uses the augmented matrix $[\mathsf{M}^{(b)}(\alpha) \mid -(\alpha^{d_b}+1)I]$ on its witness and quotient slices, with zeros elsewhere. Random-coefficient batching (Section 3.4) then checks all blocks in one sum-check over the augmented witness.

Remark 7.2 (Per-relation ring dimension). By Lemma 7.1, the ring dimension is a per-row-block choice: d_b enters only through a public weight and a degree bound, while the commitment to the flat $\mathbb{Z}_q$-coefficient vector is unchanged. Consequently, coefficientwise norm bounds and commitment binding transfer across views, and Section 4.4 may choose each commitment matrix's ring dimension independently of the fold ring constrained by the challenge set.

Applied to the active rows, Lemma 7.1 bounds the probability that a false lifted relation passes by $(2d_{\max}-1)/|E|$, with $d_{\max}$ from (107); the carrier row counts once, at dimension d_{car}.

Quotient-Free Checking via Transpose Convolution Multiplication by a public multiplier A is linear in the witness coefficients, so the verifier can reduce each basis product AX^j modulo X^d+1 itself, evaluate it at α, and use the result as the public weight of w_j. No private quotient enters the next witness.

Reduction before evaluation. Fix one row, write $d := d_r$ and $R_d := K_r[X]/(X^d+1)$, and let $P \bmod (X^d+1)$ denote the remainder of degree less than d. For $A \in R_d$ and $\alpha \in E$, the *residue kernel* is

$$\kappa^{(d)}_{A,\alpha}(j) := \Big(A(X)X^j \bmod (X^d+1)\Big)(\alpha), \qquad j \in [d]. \tag{117}$$

For $A(X) = \sum_{s=0}^{d-1} a_s X^s$, each monomial wraps around the modulus at most once, since $s+j \le 2d-2$, so

$$\kappa^{(d)}_{A,\alpha}(j) = \sum_{\substack{s\in[d]\\ s+j<d}} a_s\alpha^{s+j} - \sum_{\substack{s\in[d]\\ s+j\ge d}} a_s\alpha^{s+j-d}. \tag{118}$$

The minus sign is the wraparound $X^d = -1$; for example, $\kappa^{(4)}_{X^3,\alpha}(1) = -1$ rather than α^4. Summing these weights checks the whole ring product:

Lemma 7.3 (Transpose-convolution identity). *Let $F \subseteq E$ be a field, let $R_d := F[X]/(X^d+1)$, and let $A, W \in R_d$, with $W(X) = \sum_{j=0}^{d-1} w_j X^j$. For every $\alpha \in E$,*

$$\Big(A(X)W(X) \bmod (X^d+1)\Big)(\alpha) = \sum_{j=0}^{d-1} w_j \kappa^{(d)}_{A,\alpha}(j). \tag{119}$$

The proof is in Appendix D.2.

One ring row. Applying Lemma 7.3 to every occurrence in the normal form (110) gives the field residual

$$z_r(\alpha) := \sum_{c\in \mathsf{Occ}_r} \sum_{j=0}^{d_r-1} \kappa^{(d_r)}_{A_{r,c},\alpha}(j) w_{\mathsf{addr}_{r,c}(j)} - Y_r(\alpha), \tag{120}$$

which is the field evaluation of the ring residual Z_r. The quotient-free reduction checks $z_r(\alpha) = 0$ for every ring row of Figure 5 at its native dimension; the scalar opening row is unchanged. A wraparound recurrence computes one kernel in $O(d)$ operations without division (Appendix C.1, which also relates it to the rotation matrices of Grand Danois [57]).

Row batching. The ring and scalar rows are batched as on the quotient-lift route. Let n_{rel} be their total number and set $s_{\mathsf{rel}} = \lceil \log_2 n_{\mathsf{rel}} \rceil$. For a row-batching point $\tau_1 \in E^{s_{\mathsf{rel}}}$, let $m^{\mathsf{qf}}_{\alpha,\tau_1}(u)$ be the sum of the batched coefficients of witness entry u, and let $V^{\mathsf{qf}}_{\alpha,\tau_1}$ be the corresponding batched public target. The complete field relation is

$$\sum_{u=0}^{2^{\mu'}-1} w_u m^{\mathsf{qf}}_{\alpha,\tau_1}(u) = V^{\mathsf{qf}}_{\alpha,\tau_1}. \tag{121}$$

Every use of a witness coordinate contributes, including uses under different native dimensions; Appendix C.1 gives the coefficient vector and its evaluation at the final sum-check point. Because it tests the reduced residual itself, the reduction has a smaller evaluation degree than quotient lifting.

Theorem 7.4 (Mixed-dimension quotient-free ring checking). *Fix the outgoing witness, every address map, every public multiplier and right-hand side, and the canonical row order before sampling α. Let $d_{\max}$ be the largest active ring dimension from (107). Then the following statements hold.*

1. *If every ring row of (110) and the selected scalar opening row hold, then (121) holds for every $\alpha \in E$ and every $\tau_1 \in E^{s_{\mathsf{rel}}}$.*
2. *If some ring row is false, the probability that all ring-row residuals vanish at a uniform $\alpha \leftarrow E$ is at most $(d_{\max}-1)/|E|$.*
3. *Conditioned on a nonzero vector of evaluated ring and field residuals, its multilinear row batch vanishes at a uniform $\tau_1 \leftarrow E^{s_{\mathsf{rel}}}$ with probability at most $s_{\mathsf{rel}}/|E|$.*

Consequently, if any relation row is false, the quotient-free reduction and row batching accept with probability at most

$$\frac{d_{\max} - 1 + s_{\mathsf{rel}}}{|E|}, \tag{122}$$

before the soundness error of the fused sum-check is added. The reduction is $d_{\max}$-special sound in α.

The proof is in Appendix D.2. No step divides by $\alpha^{d_r} + 1$, so roots of the cyclotomic moduli need no rejection.

Protocol consequence. The schedule digest fixes the reduction, and the outgoing commitment is bound before the existing challenges α, τ_0, and τ_1 are sampled in their prescribed order; the fused sum-check is the same as on the quotient-lift route, and no message or challenge is added. The order matters: a row-combination challenge sampled before the witness is bound would let the prover choose a nonzero residual in the kernel of the resulting linear combination (Appendix J.1). The outgoing witness of (76) has no quotient intervals; all other components are unchanged.

7.4 The Fused Sum-Check

One sum-check over the committed witness now checks the batched relation, binds the range-check output $s_{\mathsf{claim}} = \widetilde{\mathbf{s}}(\mathbf{r}_{\mathsf{virt}})$ for the uncommitted values $\mathbf{s}(x) = \mathbf{w}(x)(\mathbf{w}(x)+1)$ (Section 6.1), and restricts the compression digits to $\{-1, 0\}$ (Figure 6). The selected ring check gives one public row weight and target. On the quotient-lift route, let

$$m_{\tau_1}(x) := \sum_i \widetilde{\mathsf{eq}}(\tau_1, i)\, \widetilde{\mathsf{M}(\alpha)}(i, x), \qquad V_{\alpha,\tau_1} := \sum_i \widetilde{\mathsf{eq}}(\tau_1, i)[\mathbf{y}(\alpha)]_i, \tag{123}$$

where $\mathsf{M}(\alpha)$ includes the quotient columns. On the quotient-free route, let

$$m_{\tau_1}(x) := \widetilde{m^{\mathsf{qf}}_{\alpha,\tau_1}}(x), \qquad V_{\alpha,\tau_1} := V^{\mathsf{qf}}_{\alpha,\tau_1}, \tag{124}$$

using (269) and (270). Both combine every row of Figure 5. The verifier samples $\zeta_{\mathsf{bin}} \leftarrow E$ when $\mathcal{I}_{\mathsf{bin}} \neq \varnothing$ (otherwise $\zeta_{\mathsf{bin}} = 0$ and $\widetilde{\mathsf{eq}}_{\varnothing} = 0$), then a distinct $\gamma \leftarrow E$, and both parties run one sum-check for

$$\begin{aligned} V_{\alpha,\tau_1} + \gamma\, s_{\mathsf{claim}} = \sum_{x \in \{0,1\}^{\mu'}} \Big[& \widetilde{\mathbf{w}}(x)\, m_{\tau_1}(x) \\ & + \Big(\gamma\, \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{virt}}, x) + \zeta_{\mathsf{bin}}\, \widetilde{\mathsf{eq}}_{\mathcal{I}_{\mathsf{bin}}}(\mathbf{r}_{\mathsf{virt}}, x)\Big) \\ & \cdot \widetilde{\mathbf{w}}(x)\big(\widetilde{\mathbf{w}}(x) + 1\big)\Big]. \end{aligned} \tag{125}$$

The binary term uses the restricted equality weights of Remark 6.2. Since the relation term has no equality factor, round messages use the standard compressed format. At the final point $\mathbf{r}_2$, the claimed value v_W and the public weights determine the summand, and the claim $\widetilde{\mathbf{w}}(\mathbf{r}_2) = v_W$ passes to the next level as $(\mathbf{r}^{(h+1)}, v^{(h+1)}) := (\mathbf{r}_2, v_W)$.

7.5 Verifier Evaluation of the Batched Row

At the final sum-check point $\mathbf{r}_x = \mathbf{r}_2$, the verifier evaluates $m_{\tau_1}(\mathbf{r}_x)$ without constructing the $2^{\mu'}$-column relation matrix: the weighted rows of $\mathsf{M}(\alpha)$ on the quotient-lift route, or the native row occurrences of (273) and the scalar rows on the quotient-free route. The non-matrix blocks are products of public factors (folding challenges, gadget powers, equality weights), which the structured evaluator of Appendix B evaluates at their witness addresses, carries included; Appendix C.3 lists the factors.

The matrix blocks read the shared setup itself. Let $\mathcal{M}_{\mathsf{setup}}$ index the active $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{D}$ views, with longest prefix $N_{\mathsf{active}} = \max_{I \in \mathcal{M}_{\mathsf{setup}}} n_I m_I d_I$. Once all weights of S_p across views and $\mathbf{B}$ slices are added into one combined weight $\bar{\omega}(p)$, the matrix blocks contribute the scan

$$\sigma_S = \sum_{p < N_{\mathsf{active}}} S_p \bar{\omega}(p). \tag{126}$$

Appendix C.4 gives the exact weights, and Appendix C.5 the resulting verifier cost.

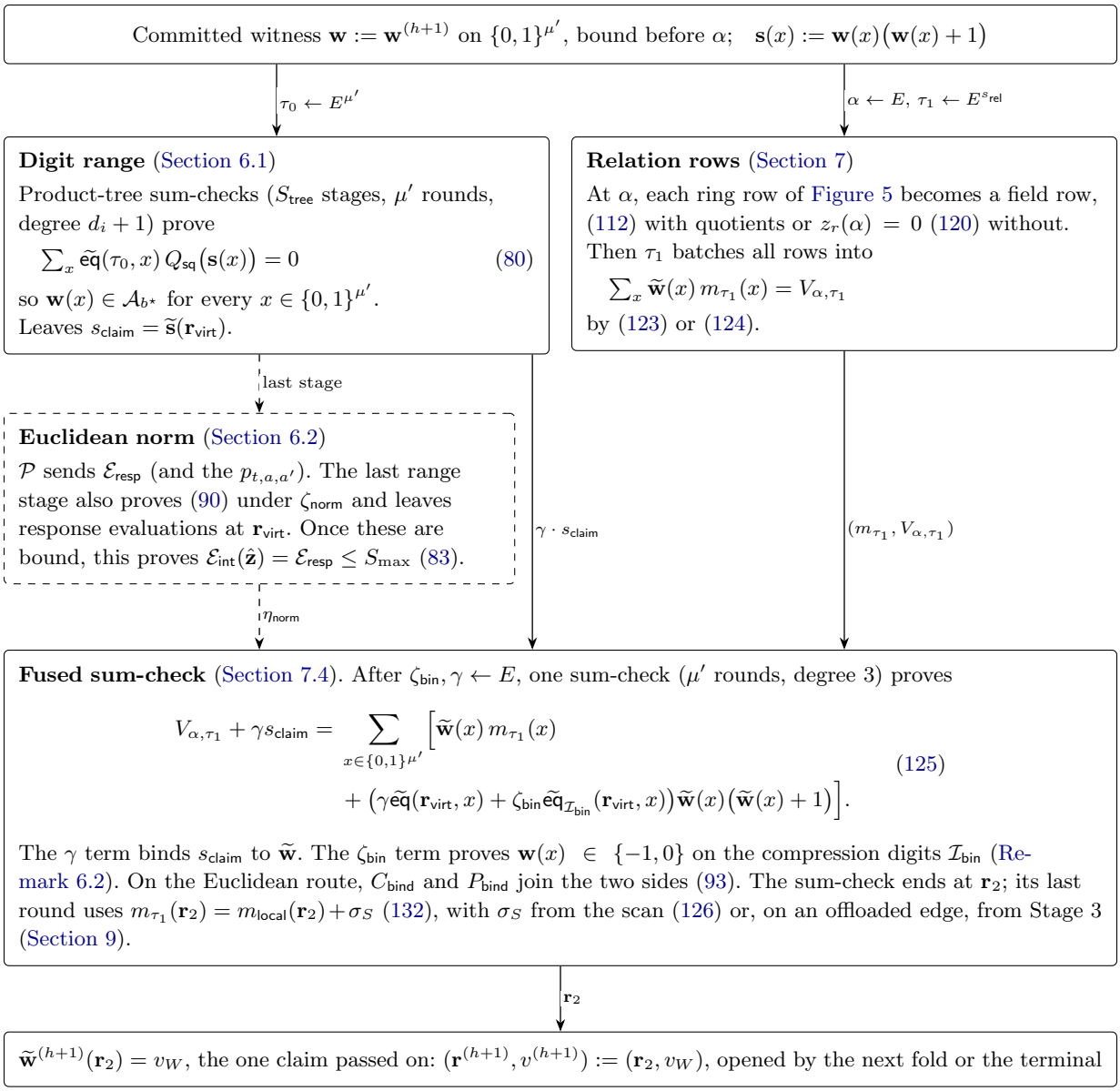


Fig. 6: The checks at one fold on the committed witness $\mathbf{w}^{(h+1)}$. Challenges: α, τ_0, and τ_1 follow the commitment in this order; the norm claims are sent before the last range stage, which then draws ζ_{pair} and ζ_{norm}; then η_{norm}, ζ_{bin}, and γ precede the fused sum-check, in this order. Dashed parts run only on the Euclidean route.

8 Recursive Opening and Terminal Handling

The fold now leaves one claim about its successor witness, regardless of how many groups it received. We can repeat the same construction as long as that successor's commitment shape agrees with the next fold's source shape. At the stopping point, we must retain the binding established by the last fold while replacing recursion by direct checks.

8.1 Composing the Folds

After each nonfinal fold h, the parties hold $\widetilde{\mathbf{w}}^{(h+1)}(\mathbf{r}^{(h+1)}) = v^{(h+1)}$ under the commitment absorbed during that fold. Any offloaded edge also carries $\mathcal{S}_h$ according to (30). The final fold instead binds the terminal inner state defined below. The *analyzed opening-mode policy* derives the opening method of each fold or terminal from its absolute level h and kind:

$$\mathsf{method}(h, \mathsf{kind}) := \begin{cases} \mathsf{SubringCoefficientPacking} & \mathsf{kind} = \mathsf{nonterminal},\ h \in \{0,1\}, \\ \mathsf{EvaluationTrace} & \text{otherwise.} \end{cases} \tag{127}$$

This rule assigns a method only to the folds present in the schedule; it does not require levels 0 or 1 to exist. Both opening methods leave an ordinary base-field witness opening, and the receiving fold performs its own opening-method reduction. Switching to evaluation trace therefore needs no bridge message, and a final packing fold can lead directly to the evaluation-trace terminal.

8.2 The Terminal Level

We stop when revealing and directly checking the remaining state costs less than another fold. Following LaBRADOR [18, §5.6], we omit outer commitments whose openings would immediately be revealed and avoid decomposing the final response. To preserve binding, we must specify what the preceding fold authenticates for these direct checks.

The state bound by the preceding fold. An ordinary edge binds the next witness through its outer commitment. At the terminal boundary, we instead need public images against which the revealed response can be checked directly. The final edge binds the canonical inner state

$$\mathbf{t}_i = \mathbf{A}_{\mathsf{term}} \mathbf{s}_i$$

for every block $i \in [B_{\mathsf{term}}]$ of its one incoming recursive-witness group, and the terminal takes this state as public input. This is what permits the terminal to omit the outer $\mathbf{B}$ relation; dropping that relation while keeping an outer commitment as the public state would be unsound. The terminal accepts no pending setup opening, so the final fold must check any incoming setup group and produce none. The final fold may use either opening method.

Reducing the incoming opening. The terminal reduces its incoming opening to one packed claim, then forms its partials and folded response. Let

$$R_{\mathsf{term}} := \mathbb{Z}_q[X]/(X^{d_{A,\mathsf{term}}} + 1).$$

The terminal's tensor reduction produces a packed claim $\bar{f}_{\mathsf{term}}(\rho_{\mathsf{term}}) = \bar{v}_{\mathsf{term}}$ and the final pair $(\theta_{\mathsf{term}}, v_{\mathsf{tensor}})$, where $v_{\mathsf{tensor}} = \theta_{\mathsf{term}} \bar{v}_{\mathsf{term}}$. Here $\theta_{\mathsf{term}} = A_\eta(\rho_{\mathsf{term}})$, using the public tensor-reduction accumulator of Section 3.7, when the final tensor-reduction relation is fused with the trace check. When the tensor reduction is the identity, $\theta_{\mathsf{term}} = 1$ and v_{tensor} is the incoming opening value. Otherwise the terminal rejects without resampling if $\theta_{\mathsf{term}} = 0$; this event precedes the response nonce and is separate from response grinding. Split $\rho_{\mathsf{term}} = (\rho_{\mathsf{blk,term}}, \rho_{\mathsf{pos,term}}, \rho_{\mathsf{pack,term}})$ as in (65), set $a_{\mathsf{term},x} := \widetilde{\mathsf{eq}}(\rho_{\mathsf{pos,term}}, x)$ and $\chi_{\mathsf{blk,term}}(i) := \widetilde{\mathsf{eq}}(\rho_{\mathsf{blk,term}}, i)$, and let $\mathcal{T}_{\rho_{\mathsf{pack,term}}}$ be the trace functional (97) at the terminal packing point. The terminal partials and folded response,

$$e_i := \langle \mathbf{a}_{\mathsf{term}}, \mathbf{f}_i \rangle \in R_{\mathsf{term}}, \qquad \mathbf{z} := \sum_{i \in [B_{\mathsf{term}}]} c_i \mathbf{s}_i \in R_{\mathsf{term}}^{m_{A,\mathsf{term}}},$$

specialize (66) and the evaluation-trace folded response.

Revealed values and direct checks. The partial opening evaluations must be fixed before the challenges that combine them. The predecessor absorbs the terminal $\mathbf{t}$ state as its outgoing binding. The terminal runs its tensor reduction, absorbs the partial opening evaluations $(e_i)_i$, sent in the clear, and then searches for a response nonce. Each candidate nonce is absorbed before the transcript derives the complete fold-challenge tuple $(c_i)_i$. A nonce is admissible only when every rejection sampler produces an accepted challenge and the resulting clear response $\mathbf{z}$ satisfies its scheduled encoding and norm checks. The prover may try at most $N_{\mathsf{try,term}}$ distinct nonces. The final transcript contains the selected nonce and absorbs $\mathbf{z}$ before the terminal checks

$$\mathbf{A}_{\mathsf{term}}\mathbf{z} = \sum_{i\in[B_{\mathsf{term}}]} c_i\mathbf{t}_i \qquad \text{in } R_{\mathsf{term}}^{n_{A,\mathsf{term}}}, \tag{128}$$

$$\sum_{i\in[B_{\mathsf{term}}]} c_i(X)e_i(X) = \mathbf{a}_{\mathsf{term}}(X)^\top \mathbf{G}_{b_{\mathsf{term}},M_{\mathsf{term}}}\mathbf{z}(X) \qquad \text{in } R_{\mathsf{term}}, \tag{129}$$

$$\theta_{\mathsf{term}} \sum_{i\in[B_{\mathsf{term}}]} \chi_{\mathsf{blk,term}}(i)\,\mathcal{T}_{\rho_{\mathsf{pack,term}}}(e_i) = v_{\mathsf{tensor}} \qquad \text{in } E. \tag{130}$$

These specialize the fold-evaluation and trace rows (103) and (98). There is no claim-batching draw for this singleton. Direct checking removes the outer $\mathbf{B}$ relation, $\mathbf{D}/\mathbf{H}$ path, quotient witness, relation sum-check, and next commitment.

Recomputing the last block. The description above sends every $\mathbf{t}_i$ and every e_i. One block of each is redundant. Write $B = B_{\mathsf{term}}$. The checks (128) and (129) are linear in $\mathbf{t}_{B-1}$ and e_{B-1}, with coefficient c_{B-1}. Every accepted fold challenge is a short nonzero element of R_{term} and hence a unit (Section 3.1), so the two relations hold exactly when

$$\begin{aligned}
\mathbf{t}_{B-1} &= c_{B-1}^{-1}\Big(\mathbf{A}_{\mathsf{term}}\mathbf{z} - \sum_{i<B-1} c_i\mathbf{t}_i\Big),\\
e_{B-1} &= c_{B-1}^{-1}\Big(\mathbf{a}_{\mathsf{term}}^\top\mathbf{G}_{b_{\mathsf{term}},M_{\mathsf{term}}}\mathbf{z} - \sum_{i<B-1} c_ie_i\Big).
\end{aligned} \tag{131}$$

The verifier can therefore compute the last block, but the omitted values must still be fixed before the challenges that test them, since (131) makes the two checks hold by construction. Let $\mathcal{H}_{\mathsf{dig}}$ be a domain-separated restriction of the random oracle with 256-bit output, applied to the canonical coefficient encoding of its argument. In place of the full state, the predecessor sends $(\mathbf{t}_i)_{i<B-1}$ and $\delta_{\mathbf{t}} = \mathcal{H}_{\mathsf{dig}}(\mathbf{t}_{B-1})$. In place of the full partials, after its tensor reduction and before the response nonce and $(c_i)_i$, the terminal sends $(e_i)_{i<B-1}$ and $\delta_e = \mathcal{H}_{\mathsf{dig}}(e_{B-1})$. After receiving $\mathbf{z}$, the verifier computes the right-hand sides of (131), rejects unless they hash to $\delta_{\mathbf{t}}$ and δ_e, and runs the trace check (130) over all B partials. It also multiplies each recomputed value back by c_{B-1} and checks (128) and (129) as written, so an incorrectly computed inverse can only cause rejection. This removes $(1+n_{A,\mathsf{term}})\,d_{A,\mathsf{term}}$ field elements from the proof and adds two 32-byte digests. Lemma 10.47 shows that the change costs at most a collision term in the random-oracle model. The standalone measurements in Section 13 use this variant; the Jolt measurements predate it.

Response bounds. The response must still satisfy the bound used by the binding argument. Since $\mathbf{z}$ is revealed, the verifier checks its scheduled coefficient-wise or squared-norm bound directly, without a digit range sum-check. The verifier rejects a nonce outside the scheduled set, a failed rejection sampler, and a response outside these bounds. Appendix L.2 specifies the response encoding.

These direct checks supply the authenticated terminal state from which Section 10 extracts backward through the folds. That analysis accounts for the terminal tensor reduction and fold challenges, and its completeness theorem includes tensor-factor rejection and exhaustion of the bounded response sampler.

9 Verifier Offloading

The verifier's remaining expensive computation is the setup scan σ_S of (126), linear in the active setup prefix. This section replaces it by a sum-check whose output is an opening of a precommitted setup prefix, which

the next fold checks beside its recursive-witness claim. The smaller $\mathbf{F}/\mathbf{H}$ maps remain local. We then define admissible schedules (Section 9.3), obtain a constant-root verifier (Section 9.4), and assemble the protocol (Section 9.5).

9.1 The Setup Claim and Setup-Product Sum-Check

Let $\mathbf{r}_2$ be the output point of the fused sum-check from Section 7.4, and let $\mathbf{r}_x$ be its projection onto the relation-address coordinates. The verifier's final row evaluation separates as

$$m_{\tau_1}(\mathbf{r}_2) = m_{\mathsf{local}}(\mathbf{r}_2) + \sigma_S. \tag{132}$$

The local term contains the structured gadget, challenge, opening-row, norm, and quotient contributions, together with the directly evaluated $\mathbf{F}/\mathbf{H}$ compression relations. The setup term σ_S contains only the contributions read from the shared setup vector through the $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{D}$ views.

The setup inner product. Recall N_{active} and the combined weight $\bar{\omega}$ from Section 7.5. Pad the setup source to length $2^{\nu_S} \geq N_{\mathsf{active}}$:

$$\mathbf{S}^\flat(p) := \begin{cases} S_p, & p < N_{\mathsf{active}}, \\ 0, & N_{\mathsf{active}} \leq p < 2^{\nu_S}. \end{cases}$$

The row-batching point τ_1, relation point $\mathbf{r}_x$, ring-reduction challenge α, and every matrix's own (n, m, d) layout determine the public coefficient weight $\omega_{\mathsf{setup}}(p)$. In particular, a view of dimension d gives flat coefficient p the quotient-lift factor $\alpha^{p \bmod d}$; the quotient-free route uses the transposed residue kernel of Section 7.3 instead. As in (126), contributions from all overlapping views add inside the same weight, and $\omega_{\mathsf{setup}}(p) := 0$ for $N_{\mathsf{active}} \leq p < 2^{\nu_S}$. Hence

$$\sigma_S = \sum_{p < 2^{\nu_S}} \mathbf{S}^\flat(p)\, \omega_{\mathsf{setup}}(p). \tag{133}$$

For every admitted offloaded edge from fold h, preprocessing commits to this canonical padded vector under the singleton source profile fixed for its successor, and the verifier setup stores the resulting public payload $C_{S,h}$ in a slot of the registry Reg selected by the schedule. The slot fixes the natural prefix length, padded commitment domain, and complete commitment profile; as part of the public setup, it cannot be supplied or replaced by the prover opening the edge.

Stage 3: the setup-product sum-check. With offloading, the prover supplies a claimed σ_S for the final check of Stage 2. Stages 1 and 2 (the range check and fused sum-check of Sections 6.1 and 7.4) are otherwise unchanged; Stage 2 also produces the next-witness opening $v_W := \widetilde{\mathbf{w}}^{(h+1)}(\mathbf{r}_2)$. Once Stage 2 fixes $\mathbf{r}_x$, the weight in (133) is determined, and Stage 3 proves the claimed value by a degree-2 sum-check for

$$\sigma_S = \sum_{p \in \{0,1\}^{\nu_S}} \widetilde{\mathbf{S}^\flat}(p)\, \widetilde{\omega}_{\mathsf{setup}}(p). \tag{134}$$

It samples a point ρ_S and ends with

$$(\text{sum-check output}) = s_{\rho_S}\, \widetilde{\omega}_{\mathsf{setup}}(\rho_S), \qquad s_{\rho_S} := \widetilde{\mathbf{S}^\flat}(\rho_S). \tag{135}$$

Its round messages use the compressed format of Table 4. The verifier evaluates the public factor $\widetilde{\omega}_{\mathsf{setup}}(\rho_S)$ from the matrix layouts, row weights, and witness addresses, without reading the setup entries or enumerating all public weights; contributions from overlapping views add, including the logical slices of $\mathbf{B}$ (Appendix C.4). The claimed value s_{ρ_S} passes to the next fold.

9.2 Passing the Setup Opening to the Next Fold

Evaluating the setup polynomial at ρ_S directly would restore the scan, so the successor receives the claim in the scheduled order

$$\mathcal{O}_{h+1} = (\mathcal{S}_h, \mathcal{W}_{h+1}), \qquad \mathcal{S}_h = (C_{S,h}, \rho_S, s_{\rho_S}), \quad \mathcal{W}_{h+1} = (C_{W,h+1}, \mathbf{r}_2, v_W). \tag{136}$$

Here $C_{S,h}$ is the registered setup commitment and $\mathcal{W}_{h+1}$ is the Stage 2 output; their points and shapes remain separate. The receiver absorbs both instances, including the schedule-selected setup slot, and binds their shared opening payload before sampling $\vartheta_{\mathsf{grp}} \leftarrow E$.

The scalar row specializes Section 7.1 to two claims. With packing, the target is

$$v_{\mathsf{boundary}} := \phi_S s_{\rho_S} + \vartheta_{\mathsf{grp}} \phi_W v_W, \tag{137}$$

where each padding factor uses its own group's domain. With evaluation trace, the shared tensor reduction first produces scaled claims $\tilde{v}_S = \theta_S \bar{s}_{\rho_S}$ and $\tilde{v}_W = \theta_W \bar{v}_W$ from the padded inputs, where $\bar{s}_{\rho_S}, \bar{v}_W$ are the packed output values and the checked nonzero factors θ_S, θ_W include the native-prefix adjustments of Section 5.2; the target is then

$$\bar{v}_{\mathsf{boundary}} := \tilde{v}_S + \vartheta_{\mathsf{grp}} \tilde{v}_W. \tag{138}$$

A fixed nonzero pair of scalar residuals cancels with probability at most $1/|E|$; two accepting transcripts with distinct coefficients force both residuals to vanish.

The receiving fold checks $\mathcal{S}_h$ before producing any new setup claim for its successor, so at most one setup opening is pending at any time.

Soundness. Binding of the setup opening is taken in the source-comparison class $C^{\mathsf{cmp}}_{S,h}$ containing the setup occurrence. This class contains both the successor's protocol occurrence and the canonical reference occurrence $\nu^0_{S,h}$ of (163), so its radius covers the mixed comparison between the extracted opening and the setup's canonical algebraic decommitment, not only two extractions under the successor profile.

Lemma 9.1 (Soundness of setup offloading). *Fix an offloaded level h. Assume that the Stage 3 product sum-check is sound and that the successor's accepted opening of the registered commitment $C_{S,h}$ is binding in the source-comparison class $C^{\mathsf{cmp}}_{S,h}$, at radius $\eta_{A,C^{\mathsf{cmp}}_{S,h}}$ of (166) for its $\mathbf{A}$ instance and the radii of (167) for its $\mathbf{B}/\mathbf{F}$ instances. Then the value σ_S used in Stage 2 equals the verifier's direct fused setup scan (126).*

The proof is in Appendix D.2.

Size of the Carried Setup Group The setup prefix is a second opening group at the next fold, and under the rank-one profile its length is bounded by that of the successor witness. The source digits of a rank-one view $I \in \mathcal{M}_{\mathsf{setup}}$ of width m_I and dimension d_I occupy $d_I m_I$ coordinates in the successor. With $|\mathbf{w}^{(h+1)}|_K$ the number of base-field coordinates of the successor witness, N_{active} of Section 7.5 specializes for $n_I = 1$ to

$$N_{\mathsf{active}} = \max_{I \in \mathcal{M}_{\mathsf{setup}}} d_I m_I \le |\mathbf{w}^{(h+1)}|_K, \tag{139}$$

so the least power-of-two prefix length $N_{\mathsf{prefix}} := 2^{\lceil \log_2 N_{\mathsf{active}} \rceil}$ satisfies

$$N_{\mathsf{prefix}} < 2N_{\mathsf{active}} \le 2|\mathbf{w}^{(h+1)}|_K. \tag{140}$$

For larger module ranks, schedule validation checks the exact rounded shape instead, and the planner uses that shape to decide which nonterminal levels offload; no fixed offloading depth is part of the protocol.

9.3 Full Schedules and Admissibility

A schedule (33) fixes every level of one protocol execution. Definition 9.2 collects the conditions for it to be admissible: the levels connect (I1–I3, C1–C6), the extractors of Section 10 apply (E1–E3), and the concrete security claim holds (S1–S3), within the analyzed family F1–F3. Admissibility is a property of the schedule, not of how a planner found it.

Scope of the analyzed family. The following restrictions simplify the implemented schedules and their analysis; the commitment and offloading identities do not require them.

(F1) *Opening and norm methods.* Nonterminal folds at levels 0 and 1, if present, use direct coefficient packing and coefficient-ℓ_∞ response bounds; later folds use evaluation trace with tensor reduction. The Euclidean route is used only for one recursive-witness response, at an evaluation-trace fold after setup openings and response chunking have ended, or at the terminal.

(F2) *Compression.* Root and setup-prefix commitments are compressed. Along the rest of the recursion, compression may stop once and does not resume.

(F3) *Ring checks.* Quotient-lift folds form a prefix, followed by an optional quotient-free suffix. Quotient-free folds occur only from level 2, use evaluation trace, and neither receive nor produce a setup opening, so their outgoing claims go directly to the next witness fold or the terminal. The terminal checks its clear response directly and has no ring-check tag.

Excluding quotient-free offloading is only a simplification. The setup-product identity (133) also holds for quotient-free rows, with the transposed residue kernel in place of the quotient-lift weights; only the public-weight evaluation cost changes.

Recursive interface.

(I1) The root source (29) satisfies the root batching conditions of Section 5, and no application-supplied group occurs at a later level.

(I2) Every later source is given by (30).

(I3) The final edge is $\mathsf{TerminalInnerState}$ and supplies the one recursive-witness state required by Section 8.2.

Compatibility conditions. For every level and every group of its source:

(C1) *Local relations.* Direct packing has $d_A = kh_{\mathsf{car}}d_{\mathsf{car}}$ and uses the same challenge in the subring consistency equation and the embedded $\mathbf{A}$ equation. A Euclidean certificate bounds the complete reconstructed response used by that equation.

(C2) *Dimensions.* The fold satisfies the matrix-dimension, coefficient-alphabet, and challenge conditions of Sections 4.1, 4.3, 4.5 and 5; every $\mathbf{B}$ slice count satisfies the slicing conditions of Section 4.5; and a multi-group source uses the shared $\mathbf{D}/\mathbf{H}$ layout of Section 5.

(C3) *Shapes.* Every target shape is exactly the source shape used at the next level.

(C4) *Certified digits.* Every committed segment uses an alphabet certified by the level that scans it. With $b^\star$ the certified range base of the scanning level, each group has $b_{1,g} \le b^\star$ for opening and inner-image digits (Remark 10.24) and $b_{z,g} \le b^\star$ for response digits (Section 5.5); quotient and compression digits satisfy their prescribed alphabets.

(C5) *Offloaded edges.* The producer's Stage 3 statement, the commitment and frozen source profile in the registry slot selected by the schedule, and the successor's setup group agree exactly. The setup group is a source occurrence at the successor, and its frozen $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{F}$ ranks meet the security floors of its comparison class under the receiving fold's certified challenge, response, and digit profiles.

(C6) *Setup groups at the receiver.* The next fold checks the setup group. At an evaluation-trace receiver, the setup and recursive-witness groups enter one shared tensor reduction under their own shapes and points, and the receiver samples fresh claim coefficients only after the shared opening payload is fixed, with the first coefficient equal to one. A fold may check one setup group and produce a new one, but setup groups do not accumulate. The final committed fold checks any incoming setup group and produces none.

Extraction conditions.

(E1) *Challenge families.* Every accepted challenge family has a certified lower bound on its cardinality and certified pairwise-difference conditions (Section 3.2).

(E2) *Samplers.* Fold challenges are drawn by rejection sampling on coordinate streams (Section 10.1). For every challenge round r and coordinate u, the schedule records the draw limit $L_{r,u}$, a certified lower bound $\underline{\alpha}_{r,u}$ on the accepted fraction, and the success bound

$$\underline{a}_{r,u} := 1 - (1 - \underline{\alpha}_{r,u})^{L_{r,u}}, \tag{141}$$

and $L_{r,u}$ and $1/\underline{a}_{r,u}$ are polynomially bounded.

(E3) *Tree size.* The checkpoint tree size (197) satisfies

$$\mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch}) \le T_{\mathsf{poly}}(\lambda, N_{\mathsf{exp},\max}), \tag{142}$$

for a fixed polynomial T_{poly}, where $N_{\mathsf{exp},\max} \ge 2^{\mu_{\max}}$ is the largest total explicit length of the public instance and the extracted witness, in base-field coordinates, over admitted schedules. The bound need not be polynomial in $\mu_{\max}$: an extractor already spends linear time to output a dense witness. The complete tree (196) is at least $\mathsf{Tree}_{\mathsf{ck}}$.

Security conditions.

(S1) *Module-SIS floor.* Every Module-SIS instance in $\mathcal{I}(\mathsf{sch})$, formed from the scheduled matrix views and certified collision bounds, meets the stated security floor. The response-derived norms are those of (78), and the instances use the source-comparison classes of (166), including their cross-profile source-path alphabet diameters.

(S2) *Honest query count.* The worst-case honest $\mathcal{H}_{\mathsf{FS}}$ query count of the transcript program, including all allowed nonce probes, is at most $Q_{\max}$.

(S3) *Fiat–Shamir budget.* The interactive error $\varepsilon_{\mathsf{int}}(\mathsf{sch})$ satisfies (195) for the recorded $(\lambda_{\mathsf{FS}}, Q_{\max})$.

Definition 9.2 (Admissible Akita schedule). *A schedule in the family F1–F3 is* extraction-admissible *if*

(i) *it satisfies the recursive interface I1–I3;*
(ii) *it satisfies the compatibility conditions C1–C6; and*
(iii) *it satisfies the extraction conditions E1–E3.*

It is admissible *if, in addition,*

(iv) *it satisfies the security conditions S1–S3.*

The extractors of Section 10 use items (i)–(iii); item (iv) is the concrete security claim, used by Theorems 10.46 and 10.52.

9.4 The Constant-Root Verifier

For each constant $\kappa_{\mathsf{root}} \ge 2$, we obtain schedules whose verifier performs $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N^{1/\kappa_{\mathsf{root}}})$ field operations (Theorem 9.3), where $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}$ and $\widetilde{\Theta}_{\kappa_{\mathsf{root}},\lambda}$ hide polylogarithmic factors in N and dependence on κ_{root} and λ.

Early-fold geometry. Unlike a balanced direct schedule, the first $\kappa_{\mathsf{root}} - 1$ folds work at the target scale

$$N_\star := \left\lceil N^{1/\kappa_{\mathsf{root}}} \right\rceil,$$

each using $B_h = \widetilde{\Theta}_{\kappa_{\mathsf{root}},\lambda}(N_\star)$ source blocks of length $M_h = \lceil N_h/B_h \rceil$, up to the required power-of-two rounding, where N_h is the total source length at level h, including any incoming setup group, in base-field coordinates. The argument uses only N_{h+1} and the regularity conditions, so it applies under either opening method; some of these folds use evaluation trace when $\kappa_{\mathsf{root}} > 3$.

Regularity conditions. Admissibility is a soundness condition and does not imply the size recurrence below. We call a family of admitted schedules *constant-root regular* if, at each of the first $\kappa_{\mathsf{root}} - 1$ levels h:

(R1) *Counts.* The numbers of source groups, claims per group, logical $\mathbf{B}$ slices, relation-row families, quotient families, and active $\mathbf{F}/\mathbf{H}$ maps are $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(1)$.

(R2) *Parameters.* Per-matrix ring dimensions, module ranks, digit depths, challenge descriptions, and finite-state address spaces are $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(1)$ as functions of the root size N.

(R3) *Other segments.* Every successor segment except the folded response has total length $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(B_h)$.

(R4) *Setup prefix.* The longest active flat $\mathbf{A}/\mathbf{B}/\mathbf{D}$ setup prefix satisfies the exact rounded balance check, with $N_{\mathsf{prefix},h} = \widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(|\mathbf{w}^{(h+1)}|_K)$.

(R5) *Local evaluations.* The exact finite-state evaluations of every remaining factored row and of $\widetilde{\omega}_{\mathsf{setup}}(\rho_S)$ take $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N_\star)$ field operations; any factor over the M_h positions is evaluated through its equality structure, without enumerating all M_h values.

These conditions concern a family as N grows, are checked on its exact successor layouts, and include any growth needed to maintain the challenge-entropy, field-error, and Module-SIS budgets; a schedule validated at one input size does not establish them.

Size recurrence. The folded response has M_h positions, the remaining data attached to the source blocks has size $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(B_h)$, and the carried setup group is bounded as in Section 9.2. Hence

$$N_{h+1} = \widetilde{O}_{\kappa_{\mathsf{root}},\lambda}\left(\frac{N_h}{N_\star} + N_\star\right). \tag{143}$$

Induction gives

$$N_h = \widetilde{O}_{\kappa_{\mathsf{root}},\lambda}\left(\frac{N}{N_\star^h} + N_\star\right).$$

Theorem 9.3 (Constant-root verifier family). *Fix a constant $\kappa_{\mathsf{root}} \geq 2$ and security parameter λ. Consider a constant-root regular family of admissible Akita schedules such that:*

(i) *the first $\kappa_{\mathsf{root}} - 1$ folds use the geometry above and setup offloading;*
(ii) *the later folds are balanced direct folds retaining the polylogarithmic parameter and per-level count bounds above, with $N_{h+1} \leq N_h^\beta$ for a fixed $0 < \beta < 1$ until the total source length first reaches a prescribed $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(1)$ threshold, where the terminal check applies; here N_h includes any incoming setup group; and*
(iii) *responses use the coefficient-ℓ_∞ route or, more generally, every selected Euclidean certificate has $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(1)$ explicit norm claims per level.*

*For a root opening of size N, the prover runs in $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N)$ operations, the proof has size $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(\log N)$, and the verifier runs in $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N^{1/\kappa_{\mathsf{root}}})$ field operations. Moreover, besides the setup seed and the short commitments authenticating the offloaded prefixes, the verifier can be provisioned with only $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N^{1/\kappa_{\mathsf{root}}})$ expanded coefficients from all active setup views (**A**, **B**, **D**, and any **F**/**H** maps).*

The proof is in Appendix F.

A larger fixed κ_{root} lowers the verifier exponent but adds early offloaded folds. A digit-expanded norm check with an unbounded number of response segments would add those claims to the proof; it is a concrete optimization outside this asymptotic family. Section 10 covers completeness and knowledge soundness for these schedules.

A Parameter Choice Attaining the Bound Lemma 9.4 gives one family meeting the conditions of Theorem 9.3. Its hardness requirement is quantitative: as the input grows, security must hold against algorithms that can already read the input, which no claim for fixed parameters provides.

Parameter-growth assumption. Write

$$L := \lambda + \lceil \log_2(N+2) \rceil + \lceil \log_2(Q_{\max}+2) \rceil.$$

For the bounds below, $\log Q_{\max} = O(\lambda + \log N)$. We use the following sufficient assumption for ordinary Module-SIS of module rank one: for each fixed constant C, one can choose
– a power-of-two degree $d \geq CL$ with $d = L^{O_C(1)}$, and
– a prime $q \equiv 5 \pmod 8$ with $\log_2 q = \Theta_C(L^2)$,
such that time-2^{CL} algorithms solve the coefficient-wise Module-SIS instances of degree d, rank one, width at most 2^{CL}, and collision bound at most 2^{CL} with probability at most 2^{-CL}. Enlarging C covers the polynomial running-time overhead of Theorem 10.46 and the allocation of error among the active maps. The assumption concerns the full SIS parameter tuple, not merely an exponential attack-cost estimate in the dimension, and adds no structure to the public matrices.

Lemma 9.4 (Existence of constant-root schedules). *Under the parameter-growth assumption above, for every fixed $\kappa_{\mathsf{root}} \geq 2$ there is a family of admissible Akita schedules satisfying Theorem 9.3. The family uses only coefficient-wise response bounds and deterministic honest-response bounds.*

The proof, and a discussion of which SIS hardness levels suffice, are in Appendix F.

9.5 The Assembled Protocol

Figure 7 specifies how the setup, commitment algorithm, and canonical fold implement the PCS interface. The security analysis of Section 10 is stated against this figure.

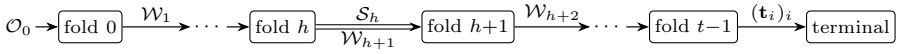
Setup($1^\lambda, \mu_{\max}$) $\to$ pp Sections 4.3 and 9.1

Fix an admissible schedule sch (Definition 9.2) with profile $(\lambda_{\mathsf{FS}}, Q_{\max})$ and setup length N_{setup} (34). From a public 256-bit seed, expand $S_i := \mathsf{XOF}_q(\mathsf{seed}; \mathsf{setup}, i)$ for $i < N_{\mathsf{setup}}$ (36); a matrix in $R_{q,d}^{n\times m}$ is the prefix view of $\mathbf{S}[0..nmd-1]$ (35). For every offloaded edge from fold h, commit to the padded prefix $\mathbf{S}^\flat$ under the singleton source profile fixed for its successor and store the payload $C_{S,h}$ in the slot of the registry Reg that sch selects. Output pp := (seed, sch, Reg).

Commit(pp, $\mathsf{shape}_{\mathsf{com}}$, $(f_p)_{p\in[P]}$) $\to$ (cm, st) Section 4.5

For every $p \in [P]$ and block $i \in [B]$, $B = \lceil N/M \rceil$: take $\mathbf{f}_{p,i}$ by (41); $\mathbf{s}_{p,i} := \mathbf{G}_{b,M}^{-1}(\mathbf{f}_{p,i})$, or $\mathbf{s}_{p,i} := \mathbf{f}_{p,i}$ for a bounded source; $\mathbf{t}_{p,i} := \mathbf{A}\mathbf{s}_{p,i}$ and $\hat{\mathbf{t}}_{p,i} := \mathbf{G}_{b_1,n_A}^{-1}(\mathbf{t}_{p,i})$. Outer image $\mathbf{u} := (\mathbf{u}_0 \| \cdots \| \mathbf{u}_{S_B-1})$, $\mathbf{u}_s := \mathbf{B}\mathbf{x}_{B,s}$ by (44) and (45). If $L_F = 2$, $\widetilde{\mathbf{u}}_0 := \mathbf{u}$, $\hat{\mathbf{u}}_j := \mathbf{G}_{\mathcal{A}_{F,j},\cdot}^{-1}(\widetilde{\mathbf{u}}_{j-1})$, $\widetilde{\mathbf{u}}_j := \mathbf{F}_j\hat{\mathbf{u}}_j$ for $j \in \{1,2\}$. Output cm := $(\mathsf{shape}_{\mathsf{com}}, \mathbf{u}_{\mathsf{pub}})$ with $\mathbf{u}_{\mathsf{pub}}$ from (46), and st: the blocks $\mathbf{s}_{p,i}$ and all digits (Definition 10.21).

Eval(pp, $\mathcal{O}_0$) $\mathcal{O}_0$ as in (29)



Preamble. Absorb the seed identity; the schedule digest, which binds $(\lambda_{\mathsf{FS}}, Q_{\max})$, every source and target shape, challenge family and sampler, nonce budget, response route, compression path, and ring-check tag; and every root handle cm_g, point $\mathbf{r}_g$, projection $\pi_{g,c}$, and claim $v_{g,c}$. $\mathcal{V}$ rejects any $\phi_{g,c} = 0$ or a projection that disagrees with its padding embedding (Section 5.1).

Folds. For $h = 0, \ldots, t-1$, run Figure 4 on $\mathcal{O}_h$ with cfg_h (31), where Open_h is packing for $h \in \{0,1\}$ and evaluation trace otherwise (127), and with edge_h (32). The next source is $(\mathcal{W}_{h+1})$ after a direct edge and $(\mathcal{S}_h, \mathcal{W}_{h+1})$ after an offloaded edge (30), whose Stage 3 runs at fold h; fold $h+1$ checks $\mathcal{S}_h$. The final edge is TerminalInnerState and leaves no setup group.

Terminal (Section 8.2). Input: the bound state $\mathbf{t}_i = \mathbf{A}_{\mathsf{term}}\mathbf{s}_i$, $i \in [B_{\mathsf{term}}]$, and the claim $\widetilde{\mathbf{w}}^{(t)}(\mathbf{r}^{(t)}) = v^{(t)}$. The tensor reduction gives $(\rho_{\mathsf{term}}, \theta_{\mathsf{term}}, v_{\mathsf{tensor}})$; $\mathcal{V}$ rejects if $\theta_{\mathsf{term}} = 0$. $\mathcal{P} \to \mathcal{V}$: $(e_i)_i$, then a nonce n from a set of size $N_{\mathsf{try,term}}$. $\mathcal{V} \to \mathcal{P}$: $(c_i)_i$. $\mathcal{P} \to \mathcal{V}$: $\mathbf{z} = \sum_i c_i\mathbf{s}_i$. $\mathcal{V}$ checks (128), (129), (130), and the scheduled coefficient-wise or squared-norm bound on $\mathbf{z}$. In the last-block variant, digests $\delta_{\mathbf{t}}, \delta_e$ replace $\mathbf{t}_{B-1}, e_{B-1}$, which $\mathcal{V}$ recomputes by (131).

Accept iff every fold of Figure 4 accepts and every terminal check holds.

Fig. 7: The Akita PCS. Root multi-group batching changes only the source $\mathcal{O}_0$; setup offloading (double arrow) changes one producer edge and the source of its immediate successor.

10 Security of Akita

This section proves completeness and knowledge soundness of the assembled protocol of Figure 7 under an admissible schedule (Definition 9.2). The analysis covers folds with one or several source instances, offloaded edges (Section 9), and public batching, which adds only the claim-combination check of Lemma 10.33. Section 10.1 proves the random-oracle facts about challenges that both proofs use: rejection sampling with a draw limit, programming a sampled challenge, and proof-of-work before algebraic challenges. Section 10.2 bounds the probability that an honest execution aborts (Theorem 10.9). Section 10.3 proves knowledge soundness, first for one fold, then for the interactive protocol of a fixed schedule (Theorem 10.40), and then for its Fiat–Shamir transform (Theorem 10.46). Section 10.4 extends both results to schedules chosen from a public family by the input size.

Shape of the main theorem. Theorem 10.46 keeps two security coordinates separate. The Module-SIS instances $\mathcal{I}(\mathsf{sch})$ of the schedule may target 128-bit quantum attack cost. The knowledge error of the Fiat–Shamir transform is a schedule-specific bound in the classical random-oracle model; QROM security is future work (Section 14). This bound charges each oracle query the error of the challenge it addresses, with each fold coordinate addressed separately (Theorem 10.17), so it is at most $\varepsilon_{\mathsf{int}}(\mathsf{sch}) + Q\,\epsilon_{\max}(\mathsf{sch})$, where $\varepsilon_{\mathsf{int}}$ is the complete interactive error. The schedule's Fiat–Shamir budget uses the coarser $(Q_{\max}+1)\varepsilon_{\mathsf{int}}(\mathsf{sch})$ (Corollary 10.48 and Section 12.2). The stated bounds take no credit for proof-of-work; Section 10.1 shows which terms it may reduce.

Route of the knowledge-soundness proof. The proof has three layers (Figure 8).

1. *One fold.* From authenticated outgoing claims, extraction recovers weak openings of the fold's incoming sources, or a short kernel vector of a scheduled matrix view (Theorem 10.31).
2. *Fixed schedule.* Composing these steps backward from the terminal gives an extractor for the interactive protocol (Theorem 10.40).
3. *Fiat–Shamir.* The opening is a suffix that proves one public linear row of evaluations of the committed vectors (Proposition 10.44). Theorem 10.45 composes it with any public-coin reduction from an application relation to such a row that is round-by-round sound for the committed vectors. Akita's public batching is the simplest such reduction (Theorem 10.46).

10.1 Challenge Sampling

Akita derives three kinds of challenges from the transcript.

– *Fold challenges:* one ring element per coordinate, sampled from an accepted family by rejection with a draw limit (Section 3.2).
– *Field vectors:* the tensor-row, range-point, and relation-row challenges, uniform over E^k.
– *Algebraic challenges at proof-of-work sites:* drawn only after a transcript-bound predicate passes.

Lemma 10.1 gives the abort probability and output law of a rejection sampler, for completeness (Definition 10.6). Lemma 10.2 and the paragraph after it let the extractors program a sampled challenge in the random-oracle model, and Proposition 10.3 bounds adaptive search at a proof-of-work site.

Lemma 10.1 (Rejection sampling with a draw limit). *Let $\mathcal{C}^{\mathsf{acc}} \subseteq \mathcal{C}_0$ be nonempty and $\alpha := |\mathcal{C}^{\mathsf{acc}}|/|\mathcal{C}_0|$. Draw at most $L \geq 1$ independent uniform elements of $\mathcal{C}_0$ and output the first one in $\mathcal{C}^{\mathsf{acc}}$, or $\perp$ if there is none. Then*

$$\Pr[\perp] = (1-\alpha)^L, \qquad \Pr[c = x \mid c \neq \perp] = \frac{1}{|\mathcal{C}^{\mathsf{acc}}|} \quad \text{for every } x \in \mathcal{C}^{\mathsf{acc}}. \tag{144}$$

For independent samplers indexed by $u \in U$, with parameters $\mathcal{C}_{0,u}$, $\mathcal{C}_u^{\mathsf{acc}}$, α_u, and L_u, all coordinates produce an output with probability

$$a := \prod_{u \in U} a_u, \qquad a_u := 1 - (1-\alpha_u)^{L_u}, \tag{145}$$

and conditioned on this event the output is uniform over $\prod_{u\in U} \mathcal{C}_u^{\mathsf{acc}}$. A certified lower bound on each α_u gives a lower bound on a.

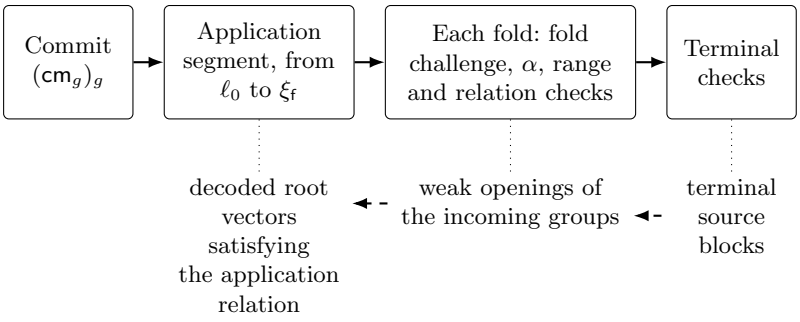


Fig. 8: Protocol order (top) and extraction order (bottom, dashed). Extraction starts at the terminal. At each fold it forks the fold challenge one coordinate at a time and recovers weak openings of the incoming groups; the ring-reduction challenge α is a checkpoint at which a candidate successor source is tested. At the root it returns the source of a root weak opening, whose decoded vectors satisfy the application relation (Proposition 10.44 and Theorem 10.45). Two different sources extracted under one commitment give a Module-SIS solution (Lemma 10.20).

Proof. The sampler outputs $\perp$ exactly when all L draws lie outside $\mathcal{C}^{\mathsf{acc}}$. For $x \in \mathcal{C}^{\mathsf{acc}}$, the first accepted draw equals x with probability

$$\sum_{j=0}^{L-1}(1-\alpha)^j \frac{1}{|\mathcal{C}_0|} = \frac{1-(1-\alpha)^L}{|\mathcal{C}^{\mathsf{acc}}|}.$$

Dividing by $\Pr[c \neq \perp]$ gives uniformity. Independence across coordinates gives (145) and the product distribution.

Coordinate streams. In the noninteractive protocol, coordinate u of a fold challenge reads its draws from its own *coordinate stream*: a lazy uniform bit stream from an ideal oracle indexed by the round context, the nonce, and u. No stream depends on the answers of another. The draw limit L_u counts complete draws from $\mathcal{C}_{0,u}$, not bit words, because decoding one draw uses rejection and reads a variable number of bits. The implementation expands each stream with SHAKE from the group's transcript seed; we analyze it in the ideal-XOF model, with the coordinate stream as the programmable object. An implementation that instead obtains each draw from a separate $\mathcal{H}_{\mathsf{FS}}$ query must count every such query in Q.

Lemma 10.2 (Programming a rejection-sampled challenge). *Let each coordinate $u \in U$ run the sampler of Lemma 10.1 on its coordinate stream, and let the transcript challenge be the output in $\prod_{u\in U}\mathcal{C}_u^{\mathsf{acc}}$ or $\perp$. Assume that decoding one draw from $\mathcal{C}_{0,u}$ and sampling the bit encoding of a prescribed element of $\mathcal{C}_{0,u}$ both take expected polynomial time. For every $c = (c_u)_{u\in U} \in \prod_u \mathcal{C}_u^{\mathsf{acc}}$ there is an exact sampler for the coordinate streams conditioned on output c, with expected work*

$$O\Big(\sum_{u\in U}\frac{L_u}{a_u}\Big)$$

draws, filter tests, and conditional encodings. Consequently, if membership in each $\mathcal{C}_u^{\mathsf{acc}}$ is efficient and every L_u and $1/a_u$ is polynomially bounded, the classical random-oracle extractor for challenges uniform on $\prod_u \mathcal{C}_u^{\mathsf{acc}}$ applies to the rejection-sampled protocol with polynomial overhead, and its coordinate-wise special-soundness denominators remain $|\mathcal{C}_u^{\mathsf{acc}}|$.

Proof. One coordinate. Fix $c \in \mathcal{C}^{\mathsf{acc}}$. Sample uniform sequences of L draws until one contains an accepted draw, and replace its first accepted draw by c. Conditioned on success, the first accepted draw is uniform on $\mathcal{C}^{\mathsf{acc}}$ and independent of its position, the rejected draws, and the unused draws. The result therefore has the distribution of a uniform draw sequence conditioned on output c. This takes $1/a$ trials in expectation, each of at most L draws. The coordinates are independent, so applying this to each gives the tuple sampler.

Bits. The stream holds bit encodings of draws, so the sampler must also produce the encoding of each draw conditioned on its value, followed by a uniform suffix; replacing a decoded value alone does not give a correctly distributed stream. For the support, invert the partial Fisher–Yates shuffle, with a uniform ordering of the support within each magnitude class. For a bounded integer below $n > 1$, sample the geometric number of rejected $\lceil \log_2 n \rceil$-bit words, each uniform in $[n, 2^{\lceil \log_2 n \rceil})$, and then the accepted word; for $n = 1$ the decoder reads no bits. Condition each sign byte on its low bit. Each step takes expected polynomial time, since integer rejection uses fewer than two words in expectation.

Programming. The extractor programs the ideal stream of the challenged coordinate, not a preimage of the SHAKE seed, and keeps all other inputs and answers. Conditioned on the accepted challenge, the prover sees the same answer distribution as in the protocol, so the reduction for uniform challenges applies with the same denominators and the added sampling work. The implementation's draw limit is 4096 per coordinate; its inner integer rejection has no limit and terminates almost surely in the ideal model.

Programming a field-vector challenge. A field-vector challenge is a uniform element of E^k answered by one oracle query. The complete tree of Theorem 10.40 forks it one coordinate at a time (Lemma 10.14): a rewind keeps the oracle key and every other table entry, fixes a prefix of the answer vector, and draws the remaining coordinates afresh. A rerun that reaches the same key therefore has the same transcript prefix, since the earlier challenges are table entries that the rewind keeps. The rewind needs an exact expected-polynomial-time sampler for the answer stream conditioned on a prefix of decoded coordinates. A base-field coordinate is decoded by rejection, so the sampler draws the rejected words and then the accepted word conditioned on its value, as in the proof above; an extension-field element has independent base-field coordinates, and the unused suffix stays uniform. With Lemma 10.2, this gives the conditional-answer sampling hypothesis of Lemma 10.14 at every stage. The Fiat–Shamir extractor of Theorem 10.46 needs it only at fold challenges, since it never samples the answer of a checkpoint or plain stage conditional on prescribed coordinates; a checkpoint retry draws those answers afresh (Proposition 10.44 and Theorem 10.45).

Response grinding is not a filter. A response-producing stage retries nonces until every sampler succeeds and every response passes its encoding and norm checks (step 2 of Figure 4). The set of accepted nonces depends on the witness, so this grind does not define a smaller fixed accepted family. Every nonce probe is an oracle query that counts toward Q, and the schedule must fit the honest grind within $Q_{\max}$. A parameter checker must not both shrink $|\mathcal{C}^{\mathsf{acc}}|$ for the grind and charge the same probes to $Q_{\max}$.

Proof-of-work at algebraic challenges. Before selected algebraic challenges, Akita requires a transcript-bound proof-of-work, separate from response grinding. The public plan fixes each protected site, its target g, and its nonce width; the instance descriptor binds the plan, and the verifier replays its nonce order and widths. At a site whose nominal error is $D_{\mathsf{alg}} 2^{-C}$, where D_{alg} is a degree or a sum of degree bounds and C is the nominal bit size of the challenge field, the target is

$$g = \max\{0,\ 128 + \lceil \log_2 D_{\mathsf{alg}} \rceil - C\}, \qquad \text{so that} \qquad 2^{-g} D_{\mathsf{alg}} 2^{-C} \le 2^{-128}. \tag{146}$$

The production profiles use $C = 128$. For $g > 0$ the prover searches at most 2^{g+7} nonces; the verifier checks a domain-separated predicate with g zero bits and then derives the algebraic challenge from an independent oracle output. A site with $g = 0$ is unprotected. The analysis uses the actual bad-challenge fraction ϵ, such as $D_{\mathsf{alg}}/|E|$, not the nominal 2^{-C}. The same mechanism serves standalone proofs and the Jolt backend (Appendix I.5).

Proposition 10.3 (Adaptive search at a protected challenge). *Fix a protected site with target g. A candidate context binds the oracle key of the transcript prefix, the site, and a nonce to one predicate/challenge pair of ideal oracle outputs, and distinct contexts have distinct, independent pairs. For each context, let B be its bad-challenge set. Assume that B is fixed before either output of the pair is first queried and, conditioned on the preceding history, has mass at most ϵ under the challenge distribution, and that the predicate passes with probability 2^{-g} independently of the challenge. Then an adaptive prover that queries either output of at most N distinct pairs and finally submits one candidate submits a passing predicate with a bad challenge with probability at most*

$$\min\{1, (N+1)2^{-g}\epsilon\}.$$

Queries may occur in either order, including before the candidate is used in the proof. The same holds for $g = 0$ with a vacuous predicate.

Proof. Enumerate the pairs in the order in which either output is first queried, and sample both outputs at that time, revealing only those queried. Conditioned on the history before this first query, the context and its bad set are fixed and the two outputs are fresh and independent, so the pair both passes and is bad with probability at most $2^{-g}\epsilon$. Later queries and the choice whether to use the pair do not enlarge this event. A union bound over at most N pairs gives $N2^{-g}\epsilon$. If the submitted candidate was never queried, its verification exposes one more fresh pair with the same bound; otherwise it was already counted. This also covers stopping early and choosing later contexts from earlier answers.

Corollary 10.4 (Several protected and unprotected sites). *For each site s of a fixed transcript plan, suppose the hypotheses of Proposition 10.3 hold with bad fraction ϵ_s, target g_s ($g_s = 0$ if unprotected), and at most N_s queried pairs. If the proof submits one candidate per site, then*

$$\Pr[\exists s : \text{ site } s \text{ passes with a bad challenge}] \le \min\Big\{1, \sum_s (N_s + 1)2^{-g_s}\epsilon_s\Big\}.$$

The sites need not be independent: each per-site bound holds conditionally on the preceding history, and a union bound applies. Each occurrence of a site in the recursive schedule is a separate site.

In particular, success probability $\delta > 0$ at a protected site requires $N + 1 \ge \delta\, 2^g/\epsilon$ queries there.

Use in the knowledge-soundness bound. A fold term of the knowledge error bounds the failure to collect sibling transcripts. This failure is not membership in a bad set fixed before the first query of a candidate, so Corollary 10.4 does not reduce it. Every other term belongs to a checkpoint or plain stage of Proposition 10.44 and Theorem 10.45, whose bad set is determined by the stage input and the fixed candidate of its checkpoint. A bad answer there has zero predicate bits and a bad challenge, as in Proposition 10.3, so a protected stage may enter the address sum (211) with its weight multiplied by 2^{-g} (Remark 10.18). Theorem 10.17 charges this weight to every queried key in any query order, so it does not need the hypothesis of Proposition 10.3 that the bad set is fixed before the first query. The budget (195) does not use this factor.

10.2 Completeness

This subsection bounds the probability that an honest execution of scheduled Akita aborts (Theorem 10.9). The schedule supplies the encoding domains, squared-norm bounds, challenge draw limits, and retry allowances $N_{\mathsf{try},h}$ of its response-producing stages, and the target and nonce allowance of each proof-of-work site. The planner of Section 12 selects response bounds with Corollary 10.10.

Every algebraic check of an honest execution holds exactly, so an abort has one of three causes: (1) at some response-producing stage, every allowed nonce attempt fails, because the challenge sampler returns $\perp$ or a response violates its bounds; (2) a required final factor of an evaluation-trace tensor reduction vanishes; or (3) a proof-of-work search exhausts its nonce allowance.

One stage at a fixed prefix. Fix an admissible schedule sch. Its response-producing stages, collected in $\mathsf{RespLev}(\mathsf{sch})$, are every nonterminal fold and the Akita terminal. For $h \in \mathsf{RespLev}(\mathsf{sch})$, let $\mathsf{Hist}_h^{\mathsf{hon}}$ be the set of reachable honest transcript prefixes immediately before the stage-h nonce is sampled; at an evaluation-trace receiver, such a prefix includes its tensor reduction and the condition that every required final factor is nonzero. A prefix $\tau \in \mathsf{Hist}_h^{\mathsf{hon}}$ fixes every source vector used at stage h.

Admission of one response. Let $\mathcal{R}_h$ index the folded responses checked at stage h: one for the core relation, one per source instance for the multi-instance relation, and one at the terminal (Corollary 11.3 gives the response-chunked specialization). For $r \in \mathcal{R}_h$, let $\mathsf{Enc}_{h,r}(\mathbf{z})$ be membership of the centered integer coefficients in the exact scheduled encoding domain, and let $T_{h,r}$ be the scheduled symmetric threshold whose coefficient cube lies in this domain. At a nonterminal fold with balanced positional base $b_{h,r}$ and depth $\delta_{f,h,r}$, put

$$M_{h,r} := \frac{b_{h,r}}{2}\frac{b_{h,r}^{\delta_{f,h,r}} - 1}{b_{h,r} - 1}, \qquad T_{h,r} := \left(\frac{b_{h,r}}{2} - 1\right)\frac{b_{h,r}^{\delta_{f,h,r}} - 1}{b_{h,r} - 1}. \tag{147}$$

Then $\mathsf{Enc}_{h,r}(\mathbf{z})$ holds exactly when every coefficient lies in $[-M_{h,r}, T_{h,r}]$, that is, when the canonical decomposition fits in $\delta_{f,h,r}$ planes. At the terminal, the domain is the one accepted by the scheduled clear prefix or fixed-width encoding. On the Euclidean route ($p_{h,r} = 2$, versus $p_{h,r} = \infty$ for the coefficient route), $S_{\max,h,r}$ is the scheduled squared-norm bound on the complete response.

Definition 10.5 (Honest response admission). *The honest admission predicate of response $r \in \mathcal{R}_h$ is*

$$\mathsf{HAdm}_{h,r}(\mathbf{z}) := \mathsf{Enc}_{h,r}(\mathbf{z}) \wedge \begin{cases} \mathsf{true}, & p_{h,r} = \infty, \\ \|\mathbf{z}\|_{2,\mathrm{coef}}^2 \le S_{\max,h,r}, & p_{h,r} = 2. \end{cases}$$

The honest prover tests admission when it selects the nonce, although it sends the norm claims later, inside Stage 1. The claims and the verifier's integer checks on them depend only on the response, not on the challenges drawn in between, so this order does not change the acceptance probability.

This predicate concerns only the honest prover. The range proof at a nonterminal fold may certify a wider alphabet than the honest base-$b_{h,r}$ one, so soundness instead uses the bounds $\beta_{\mathsf{fold}}^{\mathsf{cert}}$ (77) on accepted responses and Δ_f^{cert} (78) on their differences; neither replaces $\mathsf{Enc}_{h,r}$ here.

Failure of one nonce attempt. Let U_h index the fold-challenge coordinates derived from one candidate nonce; coordinate u samples the accepted family $\mathcal{C}_{h,u}^{\mathsf{acc}}$ by rejection from the base family $\mathcal{C}_{0,h,u}$ with draw limit $L_{h,u}$. By Lemma 10.1, a fresh candidate nonce yields $\perp$ or a tuple $\boldsymbol{c}_h$ that, conditioned on $\boldsymbol{c}_h \neq \perp$, is uniform over $\prod_{u \in U_h} \mathcal{C}_{h,u}^{\mathsf{acc}}$.

Definition 10.6 (Conditional nonce failure). *Write $\alpha_{h,u} := |\mathcal{C}_{h,u}^{\mathsf{acc}}|/|\mathcal{C}_{0,h,u}|$, so that $\alpha_{h,u} = 1$ for an unfiltered coordinate. The challenge-construction success probability is*

$$a_h := \prod_{u \in U_h} \left(1 - (1 - \alpha_{h,u})^{L_{h,u}}\right). \tag{148}$$

For $\tau \in \mathsf{Hist}_h^{\mathsf{hon}}$, the response-failure and nonce-failure probabilities are

$$q_h^{\mathsf{rsp}}(\tau) := \Pr_{\boldsymbol{c}_h}\left[\neg \bigwedge_{r \in \mathcal{R}_h} \mathsf{HAdm}_{h,r}(\mathbf{z}_{h,r}(\tau, \boldsymbol{c}_h)) \;\middle|\; \boldsymbol{c}_h \neq \perp\right], \tag{149}$$

$$q_h(\tau) := 1 - a_h\big(1 - q_h^{\mathsf{rsp}}(\tau)\big). \tag{150}$$

Thus $q_h(\tau)$ is the probability that one complete nonce attempt fails, including challenge-sampler exhaustion.

Lemma 10.7 (Completeness of one response stage). *Fix $h \in \mathsf{RespLev}(\mathsf{sch})$ and $\tau \in \mathsf{Hist}_h^{\mathsf{hon}}$. Conditioned on this prefix, an honest prover allowed $N_{\mathsf{try},h}$ distinct nonce trials fails to produce an admissible response tuple with probability*

$$q_h(\tau)^{N_{\mathsf{try},h}}.$$

Conditioned on finding an admissible tuple, all remaining honest equations of the stage hold. A nonterminal fold produces the scheduled honest successor opening, while the terminal accepts.

Proof. Distinct nonce inputs give independent runs of the same bounded challenge sampler, while τ, and hence every source vector, stays fixed. All trials therefore fail with the stated probability. For an admissible nonce, the canonical response decompositions exist at a nonterminal fold, the terminal encoding is valid at the terminal, and every scheduled Euclidean bound holds. The commitment equations, opening-method partial-evaluation equations, ring reductions, range and norm checks, fused relation sum-check, setup-product check, and recursive boundary opening are then exact identities for the honest witness, and so are the terminal checks on its revealed honest source.

A bound uniform over histories. The retries at stage h reuse the sources fixed by τ, so the power in Lemma 10.7 must be taken before averaging over τ. An average bound does not suffice: in general,

$$\mathbb{E}_\tau[q_h(\tau)] \le \delta_h \quad \not\Longrightarrow \quad \mathbb{E}_\tau\big[q_h(\tau)^{N_{\mathrm{try},h}}\big] \le \delta_h^{N_{\mathrm{try},h}}.$$

Definition 10.8 (Uniform conditional nonce-failure bound). *The schedule has a* uniform conditional nonce-failure bound $(\delta_h)_{h\in\mathsf{RespLev}(\mathsf{sch})}$ *if*

$$q_h(\tau) \le \delta_h \qquad \text{for every } h \in \mathsf{RespLev}(\mathsf{sch}) \text{ and } \tau \in \mathsf{Hist}_h^{\mathsf{hon}}. \tag{151}$$

The bound is certified *when the sampler-success probability and the response-admission probability are both bounded by proof. A* modeled-response assumption *with parameters* $(\widehat{\delta}_h^{\mathsf{rsp}})_h$ *instead asserts*

$$q_h^{\mathsf{rsp}}(\tau) \le \widehat{\delta}_h^{\mathsf{rsp}} \qquad \text{for every } h \in \mathsf{RespLev}(\mathsf{sch}) \text{ and } \tau \in \mathsf{Hist}_h^{\mathsf{hon}}. \tag{152}$$

The schedule-level bound. For each active proof-of-work site i in the schedule, let $g_i > 0$ be its target and M_i its nonce allowance, and define

$$\varepsilon_{\mathsf{pow}}(\mathsf{sch}) := \sum_{i:g_i>0} (1-2^{-g_i})^{M_i}. \tag{153}$$

A zero-bit site performs no search and cannot exhaust. The implementation uses $M_i = 2^{g_i+7}$, so each summand is at most e^{-128}. Each algebraic challenge is derived independently after the passing nonce is fixed, so proof-of-work does not change the challenge distributions used here.

Theorem 10.9 (Completeness of scheduled Akita). *Fix an admissible schedule* sch*; by Section 9.3, it is fixed independently of the expanded setup and before the protocol challenges. For each* $h \in \mathsf{RespLev}(\mathsf{sch})$*, suppose* ξ_h *satisfies*

$$\mathbb{E}\big[\mathbf{1}_{\{\mathsf{Reach}_h\}} q_h(\mathsf{Tr}_{<h})^{N_{\mathrm{try},h}}\big] \le \xi_h \tag{154}$$

for every honest opening request that passes the public admission checks. Here Reach_h *is the event that the honest execution reaches the nonce step at level* h*, and* $\mathsf{Tr}_{<h}$ *is its transcript prefix there.*

Let $\mathsf{EvalRecv}^+(\mathsf{sch})$ *be the set of evaluation-trace receivers, including the terminal when applicable, whose tensor reduction is nondegenerate. For* h *in this set, let* G_h *be its number of incoming groups,* κ_h *its tensor split parameter,* $m_{\max,h}$ *its largest tensor-reduction tail arity, and* E_h *its extension field. Then the honest prover is accepted in the random-oracle model with probability at least*

$$1 - \varepsilon_{\mathsf{comp}}(\mathsf{sch}),$$
$$\varepsilon_{\mathsf{comp}}(\mathsf{sch}) := \sum_{h\in\mathsf{RespLev}(\mathsf{sch})} \xi_h + \sum_{h\in\mathsf{EvalRecv}^+(\mathsf{sch})} \frac{G_h(\kappa_h + m_{\max,h})}{|E_h|} + \varepsilon_{\mathsf{pow}}(\mathsf{sch}). \tag{155}$$

Proof. All honest algebraic checks are exact identities. By Lemma 10.7, the probability that the execution reaches level h and exhausts its response nonces is the left-hand side of (154). At a nondegenerate evaluation-trace receiver, each of the G_h required final factors is a nonzero polynomial, jointly in the row-batching point and the common tensor-reduction point, of degree at most $\kappa_h + m_{\max,h}$. The honest-abort part of Theorem 5.1 therefore bounds its rejection probability by $G_h(\kappa_h + m_{\max,h})/|E_h|$. An identity tensor reduction has no such rejection event. At each active proof-of-work site, independent predicate outputs give exhaustion probability $(1-2^{-g_i})^{M_i}$. A union bound over these sites, response stages, and nondegenerate receivers gives (155).

Corollary 10.10 (Conditional completeness). *If the schedule has a uniform conditional nonce-failure bound* $(\delta_h)_h$ *(Definition 10.8), then Theorem 10.9 applies with* $\xi_h = \delta_h^{N_{\mathrm{try},h}}$*. In particular,*

$$\varepsilon_{\mathsf{comp}}(\mathsf{sch}) \le \sum_{h\in\mathsf{RespLev}(\mathsf{sch})} \delta_h^{N_{\mathrm{try},h}} + \sum_{h\in\mathsf{EvalRecv}^+(\mathsf{sch})} \frac{G_h(\kappa_h + m_{\max,h})}{|E_h|} + \varepsilon_{\mathsf{pow}}(\mathsf{sch}). \tag{156}$$

The bound is conditional on a modeled-response assumption only where a response bound comes from the response model.

Proof. For every reachable prefix, $q_h(\mathsf{Tr}_{<h})^{N_{\mathrm{try},h}} \le \delta_h^{N_{\mathrm{try},h}}$. Substitute this inequality into (154).

Rare exceptional histories. A uniform bound on $q_h(\tau)$ is sufficient but not necessary. Suppose that a proved set $\mathcal{G}_h$ of good prefixes satisfies

$$\Pr[\mathsf{Reach}_h \wedge \mathsf{Tr}_{<h} \notin \mathcal{G}_h] \le \eta_h \qquad \text{and} \qquad q_h(\tau) \le \delta_h \quad \text{for every } \tau \in \mathcal{G}_h.$$

Splitting the expectation in (154) according to whether $\mathsf{Tr}_{<h} \in \mathcal{G}_h$ gives $\xi_h = \eta_h + \delta_h^{N_{\mathrm{try},h}}$. Since retries keep the prefix fixed, η_h is charged once rather than raised to the retry count. A proved high-probability bound on recursive source squared norms can thus replace a deterministic one.

Obtaining the response-admission bound. Since a_h is certified combinatorially, it remains to bound $q_h^{\mathsf{rsp}}(\tau)$ at each honest prefix. Appendix K gives three routes:

1. *Deterministic envelopes* on coefficients and squared norms make every successfully sampled response fit its scheduled bounds (Corollary K.9).
2. *Tail bounds* allocate a failure probability to each response's encoding and squared-norm checks, and a union bound combines them without assuming independence between the checks or between responses (Corollary K.10). Appendix K.1 gives the source estimates for unfiltered dense and one-hot sources and for a symmetric filter, whose accepted distribution need not have independent signs.
3. *The response model* of Appendix K.3 proposes the bounds and the values $\widehat{\delta}_h^{\mathsf{rsp}}$. Its Gaussian surrogate motivates (152) but does not prove it for the actual integer response.

The first two routes are certified; all source envelopes must hold after fixing the preceding transcript, including its earlier successful nonce searches. Our optimized later-fold bounds use the third route, through the per-response premise (340). LaBRADOR and Greyhound also choose concrete response bounds from distributional estimates [18, §5.4][82, §5]; here the premise is stated for every preceding honest transcript.

On the third route, (150) and (152) bound the failure probability of one complete nonce attempt by

$$\widehat{\delta}_h^{\mathsf{tot}} := 1 - a_h(1 - \widehat{\delta}_h^{\mathsf{rsp}}), \tag{157}$$

and Corollary 10.10 with $\delta_h = \widehat{\delta}_h^{\mathsf{tot}}$ then gives (156). The modeled premise affects only the honest abort probability and expected prover work, not knowledge soundness.

10.3 Knowledge Soundness

This subsection proves Theorem 10.46 in five steps: extraction tools that do not use Akita's algebra (Theorem 10.17); a reduction from short kernel vectors of the stacked-prefix setup to ordinary Module-SIS (Lemma 10.20); extraction for one nonterminal fold, which yields a weak opening with a unique source or a short kernel vector once its outgoing claims are authenticated (Theorem 10.31); soundness of the checks after a fold challenge, with the knowledge error of each level; and the interactive and Fiat–Shamir extractors. The extraction results use only items (i)–(iii) of Definition 9.2, which make a schedule extraction-admissible. Item (iv), the Module-SIS floor and the Fiat–Shamir budget, enters only the concrete security claim of Theorem 10.46.

Transcript Trees and Checkpoint Extraction The tools below do not use Akita's algebra; they extend the coordinate-wise special-soundness (CWSS) framework of Section 3.8 in three steps: the coordinates of one challenge may come from sets of different sizes (Corollary 10.11); a vector challenge may be interpolated one coordinate at a time (Lemma 10.14); and a stage that only tests a witness already fixed by the transcript needs no rewinding, with each queried challenge address charged its own error in the extraction experiment of Definition 10.15 (Theorem 10.17). The interactive extractor of Theorem 10.40 uses the first two, and the proof of Theorem 10.46 uses the third.

Corollary 10.11 (Heterogeneous-coordinate CWSS accounting). *Let round i of a public-coin protocol sample an independent tuple from $\prod_{u \in U_i} \mathcal{X}_{i,u}$. Suppose a deterministic extractor needs $k_{i,u}$ distinct values at coordinate u, with all other coordinates fixed, and succeeds from a prefix-consistent accepting tree*

with flat coordinate-wise branching within each round. If the tree size below is polynomial in the security and explicit-instance parameters, then the interactive knowledge error and expected transcript-tree size are

$$\sum_i \sum_{u \in U_i} \frac{k_{i,u} - 1}{|\mathcal{X}_{i,u}|}, \qquad \prod_i \left(1 + \sum_{u \in U_i} (k_{i,u} - 1)\right), \tag{158}$$

respectively. The Fiat–Shamir transform is knowledge sound against a classical Q-query random-oracle adversary with the first quantity multiplied by $Q + 1$, under the conditions of Lemma 10.14(ii).

Proof. Run the counting argument of Theorem 3.10 for each coordinate separately. Failing to obtain the $k_{i,u}$ required values contributes at most $(k_{i,u} - 1)/|\mathcal{X}_{i,u}|$, and a union bound over coordinates and rounds gives the first expression. At round i the special-sound structure has $1 + \sum_u (k_{i,u} - 1)$ children, which gives the second. The Fiat–Shamir statement applies the $Q + 1$ loss to the interactive error.

In the homogeneous case $|U_i| = \ell_i$, $\mathcal{X}_{i,u} = \mathcal{X}_i$, and $k_{i,u} = k_i$, round i contributes $\ell_i(k_i - 1)/|\mathcal{X}_i|$ to the error and $\ell_i(k_i - 1) + 1$ to the tree product, as in [43, Lemma 2.31]. The heterogeneous form applies when one Akita nonce derives a domain-separated tuple of independent group challenges whose accepted families have different cardinalities.

Definition 10.12 (Nested interpolation tree). *For a challenge vector $t \in \prod_{u=1}^{a} S_u$, a nested interpolation tree of arities $(k_1, \ldots, k_a)$ branches on its coordinates in order. Below every prefix $(t_1, \ldots, t_{u-1})$ there are k_u distinct choices of t_u, each with its own complete suffix subtree. Thus the tree has $\prod_u k_u$ leaves. All leaves share the transcript preceding t; there is no intervening prover message inside this vector challenge. Each leaf has a complete accepting continuation for subsequent protocol rounds. This is different from the flat coordinate-wise sibling family in Definition 3.8.*

Lemma 10.13 (Interpolation on a nested tree). *Let $D \in E[T_1, \ldots, T_a]$ have individual degree less than k_u in T_u. If D vanishes at every leaf of a nested interpolation tree of arities $(k_1, \ldots, k_a)$, then $D = 0$. In particular, binary branching in every coordinate, with 2^a leaves, suffices for a multilinear residual.*

Proof. Induct on a. In each subtree below the root, the induction hypothesis makes the restriction to its fixed first coordinate the zero polynomial. There are k_1 such distinct first coordinates. Univariate interpolation in T_1 therefore makes every coefficient polynomial zero. The case $a = 0$ is immediate.

Lemma 10.14 (Mixed transcript trees). *Consider independent product challenges, with each protocol round designated as either a flat coordinate-wise stage or a nested interpolation stage. Suppose a deterministic extractor succeeds from every complete accepting tree with those structures in time polynomial in its size and the explicit instance length. For a stage r with coordinate sets $S_{r,u}$, let e_r be its summand in the error of (158), and let b_r be its factor in the tree size there for a flat stage and $b_r := \prod_{u \in U_r} k_{r,u}$ for a nested stage.*

1. *The interactive knowledge error is at most $\sum_r e_r$. If $\prod_r b_r$ is polynomial, the associated tree extractor has expected polynomial running time.*
2. *In the classical random-oracle model, the knowledge error of the Fiat–Shamir transform is at most $(Q + 1) \sum_r e_r$, under two conditions. First, the Fiat–Shamir input of every stage injectively encodes the stage and the prover messages up to that stage, including the statement. Second, an oracle answer can be sampled conditional on any prescribed prefix of its challenge coordinates in expected polynomial time.*

The first condition does not ask the input to encode the earlier challenges. This matches a duplex-sponge transcript such as Akita's, which absorbs the statement and the prover messages and squeezes each challenge without absorbing it. We model the sponge as a random oracle on these inputs; Chiesa and Orrù [33] relate duplex-sponge Fiat–Shamir to a transform with an independent random function for each round. The earlier challenges are then not part of a oracle input but answers of the oracle at the inputs of earlier stages, and the extractor must keep those answers when it reruns the prover (Appendix E.1). The second condition concerns the full answer distribution, including any sampler encoding, not a preimage of a concrete hash seed. A vector challenge returned by one oracle query counts as one query in the bound. The proof is in Appendix E.1.

Extraction with checkpoints. Lemma 10.14 branches at every stage, but a protocol that commits to a witness and then tests it needs less. Once the extractor has a candidate witness from the transcripts below the commitment, the tests are public-coin checks of one fixed object, and a false claim about it passes a challenge only if the challenge lies in its *bad set*, a small set determined before the challenge is drawn. At a test the extractor therefore evaluates the tested relation on the candidate instead of collecting sibling transcripts. If the relation holds, nothing more is needed. If it fails, one more extraction below the same commitment returns either a different candidate, which breaks binding, or the same failing candidate, which requires a challenge in its bad set. The tree branches only where the algebra needs several transcripts, and at most doubles at each test.

This is the commitment-predicate argument of Aardal et al. [1, Section 5 and Appendix H], extended from one predicate challenge to a sequence of challenge stages, analysed round by round [30] against the fixed candidate, and run inside the multi-round sampling game of [12].

We keep the conventions of Lemma 10.14. A protocol with stages $1, \dots, m$ has transcripts

$$(a_1, \rho_1, \dots, a_m, \rho_m, a_{m+1}),$$

where a_1 contains the statement. For $r \le m+1$, the stage-r input ξ_r is the transcript through a_r, so ξ_{m+1} is the complete transcript. The oracle does not read ξ_r itself but its *key* $\langle \xi_r \rangle$, which injectively encodes r and the messages $a_1, \dots, a_r$, as in Lemma 10.14(ii). For $r \le m$, ρ_r is the complete oracle answer at $\langle \xi_r \rangle$, distributed as Ω_r, including every coordinate of a vector challenge and any proof-of-work predicate bits that the verifier checks at that stage (Section 10.1).

A stage-r key k determines, for each $j \le r$, the stage-j key $k_{\le j}$ of its first j messages. Write $k \preceq k'$ if $k = k'_{\le j}$ for the stage j of k, and

$$\mathrm{Sub}(k) := \{k' : k \preceq k'\}$$

for the keys whose messages extend those of k, including k itself. For an oracle table J and a stage-r key k with messages $a_1, \dots, a_r$, the input *reconstructed* from J is

$$T_J(k) := \big(a_1, J(k_{\le 1}), a_2, \dots, J(k_{\le r-1}), a_r\big),$$

where $J(k_{\le j})$ is the complete stage-j answer, read from all coordinate addresses at an indexed stage. For a prover P^* with a fixed random tape and an oracle table J, A^J runs P^* and then the verifier, which queries every challenge address of the output transcript before it decides; $\xi_r(J)$ is the stage-r input of that transcript. Its challenges are table entries, so $\xi_r(J) = T_J(\langle \xi_r(J) \rangle)$.

Reconstruction and first queries. The proofs in Appendix E use two facts about message keys.

(F1) $T_J(k)$ depends only on the entries of J at the keys $k_{\le 1}, \dots, k_{\le r-1}$, which belong to stages before r. Tables that agree at every address of a stage before r thus reconstruct the same input from every stage-r key.

(F2) If J and J' differ only at one address a, then A^J and $A^{J'}$ make the same queries and receive the same answers until their first query to a. In particular, A^J queries a exactly when $A^{J'}$ does.

Both follow from the definitions: (F1) because T_J reads only those entries, and (F2) by induction over the queries. A single address can appear in several transcripts of one tree, with a different reconstructed input in each, because the extractor changes the earlier challenges between runs. For a fixed table, an address has one reconstructed input.

Challenge addresses. The oracle entries that a stage reads are its *challenge addresses*.

- An *ordinary* stage r has one address, $\langle \xi_r \rangle$, whose value is ρ_r.
- An *indexed* stage r has one address $(\langle \xi_r \rangle, u)$ for each $u \in U_r$, and $\rho_r = (\rho_{r,u})_{u \in U_r}$ with $\rho_{r,u}$ the value at $(\langle \xi_r \rangle, u)$.

The key of an address is its first component at an indexed stage and the address itself otherwise. For a key k, the *message subtree* $\mathrm{Sub}(k)$ also denotes the set of addresses whose key lies in $\mathrm{Sub}(k)$. At an indexed stage these entries are independent, the coordinate decoded from $\rho_{r,u}$ takes each value d of a set $S_{r,u}$ with probability $p_{r,u}(d) \le 1/|S_{r,u}|$, and every other outcome makes the verifier reject. The extractor needs, in expected polynomial time, an entry conditioned on a prescribed value d, and a coin with bias

$|S_{r,u}|p_{r,u}(d)$ together with, when the coin is 0, an entry conditioned on rejection; these give the representatives of Appendix E.1. The rejection-sampled coordinate streams of Lemma 10.2 have this form, with $S_{r,u}$ the accepted family: the bias is the acceptance probability of the bounded sampler, and one fresh run of that sampler supplies the coin and, when it aborts, the rejected entry. Let

$$\mathsf{Addr}(J) := \{\text{challenge addresses queried by } A^J\},$$

and let Q be the number of oracle queries made by P^*, where a query to one coordinate stream counts once.

Auxiliary environment. An *auxiliary environment* is a pair

$$\mathsf{aux} = (z, \mathcal{O}) \leftarrow \mathcal{D}_{\mathsf{aux}}$$

of an auxiliary input z and a tuple $\mathcal{O}$ of functions, drawn independently of the challenge oracle. For Akita, z contains the public setup. The prover, the verifier, the relations, and the maps of Definition 10.16 may read z and query $\mathcal{O}$. The extractor reprograms only the challenge oracle and keeps aux in every rerun. An auxiliary algorithm whose answers depend on its query history, such as a signing simulator, is part of the prover: its initial state is fixed with the prover's tape and restored at every rerun.

Definition 10.15 (Extraction experiment). *Let* R *and* $\mathsf{R}_{\mathsf{bind}}$ *be relations that may depend on* aux*. The knowledge error of an extractor* $\mathcal{E}$ *against a prover* P^* *is*

$$\kappa(P^*, \mathcal{E}) := \Pr\left[b = 1 \ \wedge \ w \notin \mathsf{R}(x) \cup \mathsf{R}_{\mathsf{bind}} \;\middle|\; \begin{array}{c} \mathsf{aux} \leftarrow \mathcal{D}_{\mathsf{aux}};\ J \text{ uniform} \\ (x, b; \mathsf{hdl}) \leftarrow A^J \\ w \leftarrow \mathcal{E}^{\mathsf{hdl}}(x) \end{array} \right].$$

Here x *is the statement output by* P^**,* b *is the verifier's decision, and the handle* hdl *fixes* aux*, the random tape and initial state of* P^**, and the oracle transcript, as in Definition 3.7.* $\mathcal{E}$ *may rerun* P^* *from that state with any challenge table and the same* aux*.*

Definition 10.16 (Checkpoint structure). *A* checkpoint structure *for a relation* R *and a binding relation* $\mathsf{R}_{\mathsf{bind}}$ *consists of the following data. All maps are deterministic, computable in polynomial time, and may read the auxiliary environment.*

(i) Stage types. *A partition* $\{1, \dots, m\} = X \sqcup H \sqcup N$ *into extraction, checkpoint, and plain stages. The* segment *of* $\ell \in H$ *is* $\mathrm{seg}(\ell) = \{\ell, \dots, \bar{\ell}\}$*, where* $\bar{\ell}+1$ *is the next stage in* $X \cup H$*, or* $\bar{\ell} = m$*. We require every plain stage to lie in a segment. Each* $r \in X$ *has the flat or nested structure of Lemma 10.14, with error* e_r *and arity* b_r*. The indexed stages form a subset* $X_{\mathsf{idx}} \subseteq X$*; each is flat with* $k_{r,u} = 2$ *for every* $u \in U_r$*, so that* $b_r = 1 + |U_r|$ *and* $e_r = \sum_{u \in U_r} 1/|S_{r,u}|$*.*

(ii) Certificates. *Sets* Y_r *for* $1 \le r \le m+1$*, where* Y_{m+1} *is the set of accepting transcripts. Every* $y \in Y_r$ *records one transcript through the message* $a_{r'}$ *of the next stage* $r' \ge r$ *in* X*, or the complete transcript if there is none;* $\xi_j(y)$ *denotes its stage-*j *input, for* $j \le r'$ *in the first case and for* $j \le m+1$ *in the second. Let* $\mathcal{T}$ *be the set of witnesses of* $\mathsf{R}_{\mathsf{bind}}$ *and* $Y_r^+ := Y_r \cup \mathcal{T}$*.*

(iii) Plain stages. $Y_r = Y_{r+1}$ *for* $r \in N$*.*

(iv) Extraction stages. *For* $r \in X$*, a map* Comb_r*. Its inputs are the families of* b_r *elements of* Y_{r+1} *that share one stage-*r *input* ξ *and whose stage-*r *answers form the flat or nested structure of that stage. Its output is an element of* Y_r *whose stage-*r *input is* ξ*, or an element of* $\mathcal{T}$*.*

(v) Checkpoint stages. *For* $\ell \in H$*, a candidate space* W_ℓ*, maps* $\mathsf{Cand}_\ell : Y_{\ell+1} \to W_\ell$ *and* $\mathsf{Good}_\ell : Y_{\ell+1} \to \{0,1\}$*, and* $Y_\ell := \{y \in Y_{\ell+1} : \mathsf{Good}_\ell(y) = 1\}$*.*

(vi) Binding. *For* $\ell \in H$ *and* $y, y' \in Y_{\ell+1}$ *with* $\xi_\ell(y) = \xi_\ell(y')$ *and* $\mathsf{Cand}_\ell(y) \ne \mathsf{Cand}_\ell(y')$*, a map* Coll_ℓ *returns* $\mathsf{Coll}_\ell(y, y') \in \mathcal{T}$*.*

(vii) Segment soundness for a fixed candidate. *For* $\ell \in H$*,* $w \in W_\ell$*,* $j \in \{\ell, \dots, \bar{\ell}+1\}$*, and every stage-*$j$ *input* ξ*, a bit* $D(\xi; w)$*; and for* $j \le \bar{\ell}$ *a set* $\mathsf{Bad}_j(\xi; w) \subseteq \mathrm{supp}\, \Omega_j$ *of probability at most* ε_j *under* Ω_j*. They satisfy:*

(a) *if* $y \in Y_{\ell+1}$ *and* $\mathsf{Good}_\ell(y) = 0$*, then* $D(\xi_\ell(y); \mathsf{Cand}_\ell(y)) = 1$*;*

(b) *if* $D(\xi; w) = 1$ *for a stage-*j *input* ξ *with* $j \le \bar{\ell}$ *and* $\rho \notin \mathsf{Bad}_j(\xi; w)$*, then* $D(\xi'; w) = 1$ *for every stage-*$(j+1)$ *input* ξ' *that extends* ξ *by the answer* ρ*;*

(c) *if* $y \in Y_{\ell+1}$*, then* $D(\xi_{\bar{\ell}+1}(y); \mathsf{Cand}_\ell(y)) = 0$.

(viii) Witness. *A map* Wit *on* Y_1 *with* $\mathsf{Wit}(y) \in \mathsf{R}(x)$ *for the statement* x *recorded in* y.

Its tree size *is* $\widehat{K} := 2^{|H|} \prod_{r \in X} b_r$*. Put* $\varepsilon_r := e_r$ *for* $r \in X$*, so that every stage has an error, and* $\varepsilon_{\mathsf{ck}} := \sum_{r=1}^{m} \varepsilon_r$*. A challenge address* a *of stage* r *has* weight

$$\epsilon(a) := \begin{cases} 1/|S_{r,u}|, & r \in X_{\mathsf{idx}} \text{ and } a = (\langle \xi_r \rangle, u), \\ \varepsilon_r, & \text{otherwise.} \end{cases} \tag{159}$$

The items follow the extractor, which works backward from stage m. A certificate $y \in Y_r$ is what it holds after stages $r, \dots, m$: a transcript prefix with what the later extraction stages recovered for it, such that every checkpoint at stage r or later passed its test. The extractor does nothing at a plain stage (item (iii)); combines b_r certificates into one by special soundness at an extraction stage (item (iv)); reads and tests the candidate at a checkpoint (item (v)), where two certificates with the same input and different candidates give a witness of $\mathsf{R}_{\mathsf{bind}}$ (item (vi)); and outputs a witness of R at the first stage (item (viii)).

Item (vii) treats a segment as a public-coin proof about one fixed candidate w, where $D(\xi; w) = 1$ means that the claims in the transcript prefix ξ are false for w. By (a), a failed test makes the claims at the checkpoint input false; (b) is round-by-round soundness, so false claims remain false unless the answer lies in the bad set; by (c), the claims at the end of the segment are true for the candidate of every certificate. Hence a certificate that fails the test has some answer on its path through the segment in a bad set of its candidate (Claim 1 in Appendix E.2). The factor $2^{|H|}$ in $\widehat{K}$ accounts for the extra extraction at each checkpoint.

Theorem 10.17 (Checkpoint extraction). *Assume a checkpoint structure whose stages in* X *satisfy the conditional-answer sampling hypothesis of Lemma 10.14. Let* Good *be an event on auxiliary environments such that the conditions of items (iv)–(viii) of Definition 10.16 hold for every* $\mathsf{aux} \in \mathsf{Good}$*. For every classical random-oracle prover* P^* *there is an extractor* $\mathcal{E}$ *with*

$$\kappa(P^*, \mathcal{E}) \le \Pr[\mathsf{aux} \notin \mathsf{Good}] + \mathbb{E}\Big[\sum_{a \in \mathsf{Addr}(J)} \epsilon(a)\Big] \le \Pr[\mathsf{aux} \notin \mathsf{Good}] + \varepsilon_{\mathsf{ck}} + Q\,\epsilon_{\max}, \tag{160}$$

where $\epsilon_{\max} := \max_a \epsilon(a)$*.* $\mathcal{E}$ *runs* P^* *at most* $\widehat{K} + (Q+1)(\widehat{K}-1) \le (Q+2)\widehat{K}$ *times in expectation. If each run, together with the work of* $\mathcal{E}$ *attached to it, has expected cost at most* T_{run} *given the history before it, then* $\mathcal{E}$ *has expected cost at most* $(Q+2)\widehat{K}\,T_{\mathsf{run}}$*. For the interactive protocol the same statements hold with* $Q = 0$*.*

Since $\epsilon_{\max} \le \max_r \varepsilon_r \le \varepsilon_{\mathsf{ck}}$, the theorem implies the familiar form

$$\kappa(P^*, \mathcal{E}) \le \Pr[\mathsf{aux} \notin \mathsf{Good}] + (Q+1)\,\varepsilon_{\mathsf{ck}}. \tag{161}$$

The first bound of (160) is sharper: it charges $N_r \varepsilon_r$ to a prover that queries N_r inputs of an ordinary stage r, and $N_{r,u}/|S_{r,u}|$ to one that reads $N_{r,u}$ streams of coordinate u of an indexed stage, whereas (161) charges $\varepsilon_{\mathsf{ck}}$ per query. The proof is in Appendix E.2.

Remark 10.18 (Work predicates and filtered challenges). The answer law Ω_r covers the whole oracle answer. Suppose a stage prepends g_r work-predicate bits that are uniform, independent of the challenge, and required by the verifier to be zero. When the predicate and the challenge are separate oracle outputs at one key, as in Akita, the pair is one answer at one address, and a query to either output counts as a query to that address. If they are not all zero, the verifier rejects every extension, so D can be kept equal to 1 and the answer need not be in the bad set. A bad answer at a checkpoint or plain stage therefore has zero predicate bits and a bad challenge, so its probability is 2^{-g_r} times that of a bad challenge alone, as in Proposition 10.3. For a challenge sampled from a fixed accepted family, ε_r is measured under that family's distribution.

Remark 10.19 (Structures selected by the statement). The stage types, arities, maps, and errors of a checkpoint structure may depend on the first message a_1, which every stage input and key contains. A node of the extractor reads them from the index tuple of its first run and keeps a later attempt only if that attempt has the same stage input. The proof of Theorem 10.17 then applies input by input, with each challenge address weighted by (159) for the structure selected by the first message that it encodes; the first bound of (160) holds with these weights.

Let $\varepsilon_{\mathsf{ck}}^{(s)}$, $\epsilon_{\max}^{(s)}$, and $\widehat{K}^{(s)}$ be the quantities of the structure selected by $a_1 = s$. Since the output transcript has one first message, the knowledge error is at most

$$\Pr[\mathsf{aux} \notin \mathsf{Good}] + \max_s \varepsilon_{\mathsf{ck}}^{(s)} + Q \max_s \epsilon_{\max}^{(s)},$$

and the expected number of runs is at most $(Q+2)\max_s \widehat{K}^{(s)}$. For the latter, each cost recurrence charges a count of queried inputs. Those with first message s contribute at most the expected number of queries of P^* with first message s plus the probability that its proof has first message s; these quantities sum to at most $Q+1$ over s.

From the Shared Setup to Module-SIS Each extractor in this section returns a witness or a short nonzero kernel vector for one matrix view of the shared setup. We collect these views and their collision bounds into an inventory $\mathcal{I}(\mathsf{sch})$ and reduce such a vector to ordinary Module-SIS (Lemma 10.20), as used by Theorems 10.40 and 10.46. The public key follows the stacked-prefix distribution of Section 4.3, not a product distribution over independent matrices.

Occurrences. Let $\mathsf{Occ}(\mathsf{sch})$ contain every scheduled source-group occurrence at a nonterminal fold and the single terminal source occurrence. An occurrence ν records its $\mathbf{A}$ matrix view and the bounds

$$K_{1,\nu},\ B_{\infty,\nu},\ K_{\mathsf{mul},2,\nu},\ B_{2,\nu},\ K_{2,\nu},\ B_{\mathsf{mul},2,\nu}$$

that its verifier checks for an extracted challenge difference and response difference; a Euclidean entry is absent when that route does not certify it.

For every registered setup commitment opened by the schedule, $\mathsf{Occ}(\mathsf{sch})$ also contains a *canonical reference occurrence* $\nu_{S,h}^0$. If $(\mathbf{s}_{S,h,c,i}^0)_{c,i}$ is the canonical algebraic decommitment fixed by preprocessing, this occurrence uses the weak certificate of Definition 10.22

$$\bar{c}_{S,h,c,i}^0 = 1, \qquad \mathbf{r}_{S,h,c,i}^0 = \mathbf{s}_{S,h,c,i}^0, \tag{162}$$

and records the schedule-certified envelopes

$$\begin{aligned} K_{1,\nu_{S,h}^0} &= 1, \\ B_{\infty,\nu_{S,h}^0} &\geq \max_{c,i} \|\mathbf{s}_{S,h,c,i}^0\|_{\infty,\mathrm{coef}}, \\ B_{2,\nu_{S,h}^0} &\geq \max_{c,i} \|\mathbf{s}_{S,h,c,i}^0\|_{2,\mathrm{coef}}. \end{aligned} \tag{163}$$

The last entry makes the first Euclidean pairing below available when the protocol occurrence certifies a Euclidean response bound. These envelopes and the canonical source-path digit alphabets depend only on the frozen setup commitment profile, not on the sampled setup values.

Comparison classes. Two nonterminal occurrence positions, including canonical reference occurrences, are *source-compatible* when the schedule assigns them the same commitment shape, including the $\mathbf{B}/\mathbf{F}$ source path, and the same $\mathbf{A}$ matrix view. The terminal occurrence is compatible with itself. These equalities partition the occurrence positions into *source-comparison classes*, independently of the commitment payload. Two extracted sources are compared only when their actual commitments are equal, and such occurrences necessarily lie in one class.

For ν, ν' in one class, define the coefficient envelope

$$\mathsf{E}_\infty(\nu, \nu') := K_{1,\nu'} B_{\infty,\nu} + K_{1,\nu} B_{\infty,\nu'}. \tag{164}$$

For the Euclidean route, let $\mathcal{P}(\nu, \nu') \subseteq \{1,2,3\}$ contain the pairings for which both occurrences certify the required bounds, and set

$$\begin{aligned} \mathsf{E}_{2,1}(\nu, \nu') &:= K_{1,\nu'} B_{2,\nu} + K_{1,\nu} B_{2,\nu'}, \\ \mathsf{E}_{2,2}(\nu, \nu') &:= K_{\mathsf{mul},2,\nu'} B_{2,\nu} + K_{\mathsf{mul},2,\nu} B_{2,\nu'}, \\ \mathsf{E}_{2,3}(\nu, \nu') &:= K_{2,\nu'} B_{\mathsf{mul},2,\nu} + K_{2,\nu} B_{\mathsf{mul},2,\nu'}. \end{aligned} \tag{165}$$

Table 3: The MSIS instances and their norm bounds for schedule sch. For each source-comparison class, the schedule selects one norm for the $\mathbf{A}$ matrix; its radius is the maximum mixed occurrence-pair envelope of (166), whose diagonal cases are the usual coefficient and Euclidean formulas. A class containing a registered setup opening also contains its canonical reference profile (163). The $\mathbf{B}$ and $\mathbf{D}$ bounds are the certified range-check alphabet diameters of Remark 10.24. Neither norm route may use the honest concentration threshold $t^\star$ of Section 5.5.

Instance I	Description	d_I	n_I	m_I	p_I	Collision Bound η_I
$\mathbf{A}^{(C)}$	Nonterminal Source-Class Inner Commitment	d_A	n_A	m_A	p_C	$\eta_{A,C}$ in (166)
$\mathbf{A}^{(\mathsf{term})}$	Terminal Inner Commitment	d_A	n_A	m_A	∞ or 2	(181) or (182)
$\mathbf{B}^{(C)}$	Source-Class Sliced Outer Commitment	d_B	n_B	m_B	∞	$\eta_{B,C}$
$\mathbf{D}^{(h)}$	Unsliced Opening Commitment	d_D	n_D	m_D	∞	$\eta_{D,h}$
$\mathbf{F}_\ell^{(C)}$	Source-Class Outer Chain Map $\ell \in \{1,\dots,L_F\}$	$d_{F,\ell}$	$n_{F,\ell}$	$m_{F,\ell}$	∞	1
$\mathbf{H}_\ell^{(h)}$	Opening-Local Chain Map $\ell \in \{1,\dots,L_H\}$	$d_{H,\ell}$	$n_{H,\ell}$	$m_{H,\ell}$	∞	1

Width of $\mathbf{B}$. The length m_B is the width of the single matrix that every slice reuses. It is not the total logical input length. This width suffices for binding. A nonzero kernel vector of the full stacked map has a nonzero slice, and padding that slice with zeros gives a kernel vector of the width-m_B matrix (Lemma 4.3).

Width of $\mathbf{D}$. The matrix $\mathbf{D}$ is not sliced. Its width m_D is the full width of its ordered partial-evaluation input.

Bounds for $\mathbf{B}$ *and* $\mathbf{D}$. For a source class C, $\eta_{B,C}$ is the maximum cross-profile diameter of (167). The bound $\eta_{D,h}$ is local to one opening occurrence. It is the diameter $b^\star - 1$ of the balanced alphabet certified at level h.

Bounds for $\mathbf{F}$ *and* $\mathbf{H}$. Every compression map has input alphabet $\{-1, 0\}$ under all scanning profiles. The reason is that the fused binary constraint certifies the complete input span of these maps (Remark 6.2). Hence $\eta_{F,C,\ell} = \eta_{H,h,\ell} = 1$.

Each comparison class C selects one norm p_C, and its exact scheduled $\mathbf{A}$ collision bound is

$$\eta_{A,C} := \begin{cases} \max\limits_{\nu,\nu' \in C} \mathsf{E}_\infty(\nu,\nu'), & p_C = \infty, \\ \max\limits_{\nu,\nu' \in C} \min\limits_{r \in \mathcal{P}(\nu,\nu')} \mathsf{E}_{2,r}(\nu,\nu'), & p_C = 2. \end{cases} \tag{166}$$

The Euclidean choice is admissible only if $\mathcal{P}(\nu,\nu') \neq \varnothing$ for every ordered pair in C. The diagonal pair $\nu = \nu'$ covers two extractions from the same occurrence.

For a $\mathbf{B}$ or $\mathbf{F}$ source-path view K shared by a class, let $\mathcal{D}_{K,\nu}$ be its verifier-certified input alphabet at protocol occurrence ν, including all authenticated coordinates; at a canonical reference occurrence it is the canonical input alphabet fixed by the registered commitment profile. Define

$$\eta_{K,C} := \max_{\nu,\nu' \in C} \max_{x \in \mathcal{D}_{K,\nu},\, y \in \mathcal{D}_{K,\nu'}} \|x - y\|_{\infty,\mathrm{coef}}. \tag{167}$$

The instance inventory. The Module-SIS instances of an admissible schedule sch form the multiset $\mathcal{I}(\mathsf{sch})$ containing:

(i) for every source-comparison class, its $\mathbf{A}$ instance with bound (166) and, for a nonterminal class, its $\mathbf{B}/\mathbf{F}$ source-path instances with bound (167); the terminal class contributes only its $\mathbf{A}$ instance;

(ii) for every nonterminal level, the shared $\mathbf{D}$ instance and every active $\mathbf{H}$ opening-chain instance, at their verifier-certified occurrence-local digit-collision norms.

Setup openings are ordinary source occurrences and enter through (i). Each $I \in \mathcal{I}(\mathsf{sch})$ records the matrix view, its ring dimension d_I, module rank n_I, length m_I, norm choice $p_I \in \{2, \infty\}$, and collision bound η_I (Table 3).

Schedule families. The computational step fixes a finite public family $\mathcal{S}$ of admissible schedules independently of the setup; a fixed schedule is the case $|\mathcal{S}| = 1$. Its inventory is

$$\mathbb{I}_{\mathcal{S}} := \bigcup_{\mathsf{sch} \in \mathcal{S}} \mathcal{I}(\mathsf{sch}),$$

where instances with different matrix views remain distinct even when their dimensions agree. For $I \in \mathbb{I}_{\mathcal{S}}$, the advantage of an algorithm $\mathcal{B}_I$ against $\mathsf{MSIS}^{(p_I)}_{q,d_I}(n_I, m_I, \eta_I)$ (Definition 3.2) is

$$\mathsf{Adv}_I(\mathcal{B}_I) := \Pr\left[0 \neq \mathbf{x},\ \mathbf{M}^\star\mathbf{x} = 0,\ \|\mathbf{x}\|_{p_I,\mathrm{coef}} \le \eta_I \;\middle|\; \begin{array}{c}\mathbf{M}^\star \leftarrow R_{q,d_I}^{n_I \times m_I} \\ \mathbf{x} \leftarrow \mathcal{B}_I(\mathbf{M}^\star)\end{array}\right].$$

Lemma 10.20 (Shared-setup collisions reduce to ordinary MSIS). *Let $\mathcal{B}$ receive a uniform flat setup vector $\mathbf{S} \leftarrow \mathbb{Z}_q^{N_{\mathsf{setup}}}$ and auxiliary data $(z, \mathcal{O})$ sampled independently of $\mathbf{S}$, halt with probability one in expected time at most $T_{\mathcal{B}}$, and output a pair $(I, \mathbf{x})$. It may choose a schedule in $\mathcal{S}$ after seeing $\mathbf{S}$. Let $\mathsf{Adv}_{\mathsf{sp}}(\mathcal{B})$ be the probability that*

$$I \in \mathbb{I}_{\mathcal{S}}, \quad 0 \neq \mathbf{x} \in R_{q,d_I}^{m_I}, \quad \mathbf{M}_I(\mathbf{S})\mathbf{x} = 0, \quad \|\mathbf{x}\|_{p_I,\mathrm{coef}} \le \eta_I,$$

where $\mathbf{M}_I(\mathbf{S})$ is the prefix view of I. Let $T_{\mathsf{embed}}(I)$ be the cost of sampling the setup coordinates outside that view and checking the output; it is linear in the setup length.

(i) *There are algorithms $(\mathcal{B}_I)_{I \in \mathbb{I}_{\mathcal{S}}}$ of expected running time at most $T_{\mathcal{B}} + T_{\mathsf{embed}}(I)$ with*

$$\mathsf{Adv}_{\mathsf{sp}}(\mathcal{B}) \le \sum_{I \in \mathbb{I}_{\mathcal{S}}} \mathsf{Adv}_I(\mathcal{B}_I). \tag{168}$$

(ii) *For every $t > 0$ there are algorithms $(\mathcal{B}_I^{(t)})_{I \in \mathbb{I}_{\mathcal{S}}}$ of strict running time at most $t + T_{\mathsf{embed}}(I)$ with*

$$\mathsf{Adv}_{\mathsf{sp}}(\mathcal{B}) \le \frac{T_{\mathcal{B}}}{t} + \sum_{I \in \mathbb{I}_{\mathcal{S}}} \mathsf{Adv}_I(\mathcal{B}_I^{(t)}). \tag{169}$$

Proof. Fix $I \in \mathbb{I}_{\mathcal{S}}$. Given its challenge $\mathbf{M}^\star$, $\mathcal{B}_I$ samples a flat setup vector $\mathbf{S}$ subject only to $\mathbf{M}_I(\mathbf{S}) = \mathbf{M}^\star$, filling every other coordinate uniformly in $\mathbb{Z}_q$. This is efficient because every matrix view is a coefficient-wise prefix of $\mathbf{S}$, with row-major coefficients fastest (Equation (35)), and $\mathbf{S}$ is exactly uniform because $\mathbf{M}^\star$ is. It then samples $(z, \mathcal{O})$ independently of $\mathbf{S}$, runs $\mathcal{B}$, and returns $\mathbf{x}$ if $\mathcal{B}$ names I with a valid vector. Its success probability is the probability that $\mathcal{B}$ succeeds and names I, whatever schedule $\mathcal{B}$ selected; the argument neither conditions on that selection nor needs independence between views. Summing over I gives (168).

For (ii), let $\mathcal{B}'$ run $\mathcal{B}$ for at most t steps and output $\perp$ otherwise. By Markov's inequality $\mathcal{B}$ exceeds t steps with probability at most $T_{\mathcal{B}}/t$, so $\mathsf{Adv}_{\mathsf{sp}}(\mathcal{B}) \le T_{\mathcal{B}}/t + \mathsf{Adv}_{\mathsf{sp}}(\mathcal{B}')$. Applying the construction of (i) to $\mathcal{B}'$ gives (169).

A sum of negligible advantages over a finite family is negligible only when the inventory has polynomial size. A concrete claim instead evaluates the sum, as in Appendix G.

Relaxed Source Extraction and Uniqueness This subsubsection proves the extraction step for one nonterminal fold (Theorem 10.31). Its output is a weak opening (Definition 10.22), whose source blocks are short only after multiplication by a short unit, and whose source projection is unique up to a matrix collision (Corollary 10.25). Theorem 10.40 iterates the step backward from the terminal (Proposition 10.29); the Fiat–Shamir proof uses Corollary 10.32.

Folding rings. Fix a nonterminal fold level h. For each source instance g, let S_g be its folding ring and let

$$\iota_g : S_g \longrightarrow R_{q,d_{A,g}}$$

be the map through which a folding challenge acts on the ambient source. The schedule fixes the opening method:

- coefficient packing for $h \in \{0, 1\}$: S_g is the challenge subring, ι_g the embedding of (38), and $\mathcal{L}_g$ and $\mathcal{V}_{g,c}$ the packing partial and scalar-opening maps of Section 5.3;
- evaluation trace for $h \ge 2$: S_g is the full evaluation-trace folding ring, ι_g the identity, $\mathcal{L}_g$ the evaluation-trace partial map, and $\mathcal{V}_{g,c}$ its public, tensor-reduction-scaled trace functional.

Below, the product of a challenge c with an ambient source means multiplication by $\iota_g(c)$.

Both methods produce the same successor claim: an ordinary evaluation claim on a multilinear polynomial with base-field Boolean evaluations, bound by an ordinary outer commitment or, at the last nonterminal fold, by the canonical inner t state of the terminal. No edge produces a tensor-reduction output claim; an evaluation-trace receiver or the terminal performs the tensor reduction on its own input.

Definition 10.21 (Akita algebraic decommitment). *Fix one source commitment*

$$\mathsf{cm}_g := (\mathsf{shape}_{\mathsf{com},g}, \mathbf{u}_{\mathsf{pub},g}). \tag{170}$$

Its Src *component fixes the ordered source-vector layouts and decoder, and the full shape fixes the block geometry, matrix views, slice maps, digit alphabets, and active compression path (Section 5.1). An* Akita algebraic decommitment $\mathsf{st}^{\star}_{\mathsf{com},g}$ *of* cm_g *to the ordered decoded commitment-grid vectors* $(f^{\star}_{g,c})_c$ *is the commitment-side part of that construction: source blocks* $\mathbf{s}_{g,c,i}$*, inner-image digits* $\hat{\mathbf{t}}_{g,c,i}$*, and* $\mathbf{B}/\mathbf{F}$ *chain openings such that, for every applicable* c *and* i*,*

(i) *the source blocks decode to the vectors and match the inner-image digits:*

$$f^{\star}_{g,c} = \mathsf{Dec}_{\mathsf{shape}_{\mathsf{com},g}}((\mathbf{s}_{g,c,i})_{i\in[B_g]}), \qquad \mathbf{A}_g\mathbf{s}_{g,c,i} = \mathbf{G}_{b_{1,g},n_{A,g}}\hat{\mathbf{t}}_{g,c,i};$$

(ii) *the prescribed slice map places the digits* $\hat{\mathbf{t}}_{g,c,i}$ *in the inputs of (44), with* $P = n_{\mathsf{clm},g}$ *and* $B = B_g$*, and the resulting* $\mathbf{B}_g$ *rows and active* $\mathbf{F}$ *chain satisfy (47), or (48) at depth zero, and terminate at* $\mathbf{u}_{\mathsf{pub},g}$*;*

(iii) *every inner-image and compression-chain digit lies in the alphabet fixed by the commitment contract.*

The decoder imposes no zero condition on an algebraic-padding cell included in the commitment. Write

$$\mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{com}}(\mathsf{pp}, \mathsf{cm}_g, (f^{\star}_{g,c})_c, \mathsf{st}^{\star}_{\mathsf{com},g}) = 1 \tag{171}$$

when these conditions hold.

The single-polynomial specialization of this relation instantiates $\mathsf{Rel}_{\mathsf{com}}$ in Definition 3.5; the group form is used by Akita's batch theorem. The relation contains the honest commitment state but requires neither the canonical gadget decomposition of $\mathbf{s}_{g,c,i}$ nor a decoded vector in the image of an honest padding map. It excludes partial opening evaluations, the $\mathbf{D}/\mathbf{H}$ path, the opening point, and the claimed value, which may change when the same commitment is opened at another point.

The batch relation. For an application batch, write its public statement as

$$\mathsf{x}_{\mathsf{batch}} := \left(\mathsf{sch}; \left(\mathsf{cm}_g, \mathbf{r}_g, (\pi_{g,c}, v_{g,c})_{c\in[n_{\mathsf{clm},g}]}\right)_{g\in[G]}\right). \tag{172}$$

An *Akita batch witness* consists of decoded full commitment-grid vectors $(f^{\star}_{g,c})_{g,c}$ and one algebraic decommitment state $\mathsf{st}^{\star}_{\mathsf{com},g}$ for every group occurrence. Define

$$\begin{aligned}&\mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{batch}}(\mathsf{pp}, \mathsf{x}_{\mathsf{batch}}; (f^{\star}_{g,c})_{g,c}, (\mathsf{st}^{\star}_{\mathsf{com},g})_g) = 1\\ &\quad\iff \bigwedge_{g\in[G]} \mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{com}}(\mathsf{pp}, \mathsf{cm}_g, (f^{\star}_{g,c})_c, \mathsf{st}^{\star}_{\mathsf{com},g}) = 1 \ \wedge \bigwedge_{g,c} \widetilde{f^{\star}_{g,c}}(\mathbf{r}_g) = \phi_{g,c}v_{g,c}.\end{aligned} \tag{173}$$

The last conjunct is the padded claim that the protocol verifies; the relation omits the padding-image predicate $f^{\star}_{g,c} = \mathsf{Pad}_{g,c}(F_{g,c})$, which would recover the natural-domain claim $\widetilde{F_{g,c}}(\pi_{g,c}(\mathbf{r}_g)) = v_{g,c}$ by (51). If one commitment occurs in several groups, Theorem 10.40 shows that their vector families agree or returns a matrix collision.

The fixed folding statement. Fix one nonterminal fold with the G ordered source groups and public data of Sections 5.1 and 7.2, which fix $(\mathsf{cm}_g)_{g\in[G]}$, the shared opening payload $\mathbf{v}_{\mathsf{pub}}$, all active matrix chains, and the scalar row. In packing mode that row uses the application weights and target; in evaluation-trace mode it uses the post-payload weights and reduced target of (101).

Definition 10.22 (Weak opening for one fold). *A* weak opening *of the fixed folding statement contains* $(\mathbf{s}_{g,c,i}, \hat{\mathbf{t}}_{g,c,i}, \hat{\mathbf{e}}_{g,c,i}, \bar{c}_{g,c,i})$ *for every* (g, c, i) *and all active compression-chain openings. It uses the statement's layouts and commitment-opening rows, relaxed for extraction by the factors* $\bar{c}_{g,c,i}$*, and satisfies:*

(i) Weak certificates. *Each* $\bar{c}_{g,c,i}$ *is a unit in* S_g *and, for fixed challenge- and response-difference bounds, satisfies*

$$\|\iota_g(\bar{c}_{g,c,i})\|_1 \le \bar{\kappa}_{1,g}, \qquad \|\iota_g(\bar{c}_{g,c,i})\mathbf{s}_{g,c,i}\|_{\infty,\mathrm{coef}} \le \bar{\beta}_{\infty,g}.$$

On a Euclidean route it also satisfies

$$\|\iota_g(\bar{c}_{g,c,i})\|_{\mathsf{mul},2} \le \bar{\kappa}_{\mathsf{mul},2,g}, \qquad \|\iota_g(\bar{c}_{g,c,i})\mathbf{s}_{g,c,i}\|_{2,\mathrm{coef}} \le \bar{\beta}_{2,g}.$$

(ii) Source. *The source blocks, inner-image digits, and* $\mathbf{B}/\mathbf{F}$ *openings satisfy Definition 10.21.*

(iii) Opening path. *The scheduled shared* $\mathbf{D}/\mathbf{H}$ *path authenticates the ordered vector*

$$\hat{\mathbf{e}} := \mathsf{concat}(\hat{\mathbf{e}}_0, \dots, \hat{\mathbf{e}}_{G-1})$$

against $\mathbf{v}_{\mathsf{pub}}$*, using* (106) *and the active compression rows or the depth-zero row* (48)*.*

(iv) Partial evaluations and scalar row. *Recomposing the partial digits in base* $b_{1,g}$ *and applying the selected mode's fixed decoding from base-field coordinates (Equations* (58) *and* (66)*) gives* $e_{g,c,i}$*, and the remaining rows enforce*

$$e_{g,c,i} = \mathcal{L}_g(\mathbf{G}_{b_g,M_g}\mathbf{s}_{g,c,i}), \qquad v_{\mathsf{fold}} = \sum_{g\in[G]}\sum_{c\in[n_{\mathsf{clm},g}]} \vartheta_{g,c}\,\mathcal{V}_{g,c}((e_{g,c,i})_{i\in[B_g]}),$$

where v_{fold} *is* v_{root} *under coefficient packing and* $\bar{v}_{\mathsf{root}}$ *under evaluation trace.*

(v) Alphabets. *Every digit vector lies in the alphabet certified for its segment.*

For $G = 1$ *this is the ordinary core relation.*

The extracted source. The *source projection* $\mathsf{Src}(\omega)$ of a weak opening ω consists of exactly its source blocks, inner-image digits, and $\mathbf{B}/\mathbf{F}$ chain openings; by condition (ii) it is an Akita algebraic decommitment. The weak certificates $\iota_g(\bar{c}_{g,c,i})\mathbf{s}_{g,c,i}$ make it unique up to a matrix collision: Lemma 10.23 compares two weak openings of one fold, and Corollary 10.25 compares occurrences opened under different profiles.

Lemma 10.23 (Weak binding for one fold). *Consider two weak openings for the same public fold instance and payloads. If they differ on any semantic block witness* $\mathbf{s}_{g,c,i}$ *or carried digit vector, then one can deterministically output a short nonzero kernel vector for* $\mathbf{A}_g$*,* $\mathbf{B}_g$*,* $\mathbf{D}$*, or one of their compression maps. A collision other than for* $\mathbf{A}_g$ *is bounded by the diameter of the corresponding verifier-certified alphabet. For* $\mathbf{A}_g$*, fix a coordinate on which the openings differ, set* $\mathbf{r}_i := \iota_g(\bar{c}_i)\mathbf{s}_i$ *and* $\mathbf{r}'_i := \iota_g(\bar{c}'_i)\mathbf{s}'_i$*, and read each norm of* $\bar{c}_i$ *below after applying* ι_g*. The kernel vector* $\mathbf{z}_A$ *satisfies all of the following bounds:*

$$\|\mathbf{z}_A\|_{\infty,\mathrm{coef}} \le \|\bar{c}'_i\|_1\|\mathbf{r}_i\|_\infty + \|\bar{c}_i\|_1\|\mathbf{r}'_i\|_\infty, \tag{174}$$

$$\|\mathbf{z}_A\|_{2,\mathrm{coef}} \le \min\left\{\begin{array}{l}\|\bar{c}'_i\|_1\|\mathbf{r}_i\|_2 + \|\bar{c}_i\|_1\|\mathbf{r}'_i\|_2,\\ \|\bar{c}'_i\|_{\mathsf{mul},2}\|\mathbf{r}_i\|_2 + \|\bar{c}_i\|_{\mathsf{mul},2}\|\mathbf{r}'_i\|_2,\\ \|\bar{c}'_i\|_2\|\mathbf{r}_i\|_{\mathsf{mul},2} + \|\bar{c}_i\|_2\|\mathbf{r}'_i\|_{\mathsf{mul},2}\end{array}\right\}. \tag{175}$$

Consequently, uniform bounds $(\bar{\kappa}_1, \bar{\kappa}_{\mathsf{mul},2}, \bar{\kappa}_2)$ *on the three challenge-difference norms and* $(\bar{\beta}_\infty, \bar{\beta}_2, \bar{\beta}_{\mathsf{mul},2})$ *on the corresponding response-difference norms give*

$$\|\mathbf{z}_A\|_{\infty,\mathrm{coef}} \le 2\bar{\kappa}_1\bar{\beta}_\infty, \tag{176}$$

$$\|\mathbf{z}_A\|_{2,\mathrm{coef}} \le 2\min\left\{\bar{\kappa}_1\bar{\beta}_2, \bar{\kappa}_{\mathsf{mul},2}\bar{\beta}_2, \bar{\kappa}_2\bar{\beta}_{\mathsf{mul},2}\right\}. \tag{177}$$

If the first and second openings instead have the occurrence profiles ν *and* ν'*, respectively, the right-hand side of* (174) *is at most* $\mathsf{E}_\infty(\nu, \nu')$*, and each available line of* (175) *is at most the corresponding* $\mathsf{E}_{2,r}(\nu, \nu')$ *from* (165)*.*

Proof. Digit paths. Apply Lemma 4.3 to every $\mathbf{B}_g/(\mathbf{F}_{g,j})_j$ chain and to the shared $\mathbf{D}/\mathbf{H}$ chain. A difference in carried digits gives the corresponding matrix collision with the stated alphabet diameter.

Source blocks. Assume therefore that all carried digit vectors agree, in particular $\hat{\mathbf{t}}_{g,c,i} = \hat{\mathbf{t}}'_{g,c,i}$ and $\hat{\mathbf{e}}_{g,c,i} = \hat{\mathbf{e}}'_{g,c,i}$ for every coordinate. If the two weak openings still differ, choose (g, c, i) with $\mathbf{s}_{g,c,i} \neq \mathbf{s}'_{g,c,i}$ and omit these indices below. The inner consistency equations give $\mathbf{A}_g\mathbf{s}_i = \mathbf{G}_{b_{1,g},n_{A,g}}\hat{\mathbf{t}}_i = \mathbf{A}_g\mathbf{s}'_i$. Define

$$\mathbf{z}_A := \iota_g(\bar{c}'_i)\big(\iota_g(\bar{c}_i)\mathbf{s}_i\big) - \iota_g(\bar{c}_i)\big(\iota_g(\bar{c}'_i)\mathbf{s}'_i\big).$$

Both factors are needed because the openings may come from branches with different transcript prefixes, so $\bar{c}_i$ and $\bar{c}'_i$ need not agree; this gives the two summands in every bound. Since $\iota_g(\bar{c}_i)$ and $\iota_g(\bar{c}'_i)$ are units and $\mathbf{s}_i \neq \mathbf{s}'_i$, the vector $\mathbf{z}_A$ is nonzero, and $\mathbf{A}_g\mathbf{z}_A = 0$.

Norm bounds. Applying (18) to each summand gives (174), and applying the three Euclidean inequalities of (19) term by term gives the three lines of (175). The uniform bounds for both openings yield (176) and (177), and the occurrence profiles of ν and ν' yield the envelope bounds.

Remark 10.24 (Certified versus honest digit alphabets). The collision norms above use the alphabet that the protocol *enforces*, not the one the honest prover uses. Honest carried digits $\hat{\mathbf{t}}, \hat{\mathbf{e}}$ are balanced base-b_1 gadget digits, but a cheating prover is constrained only by the digit range check of the level that scans them (Section 6.1), or by the terminal wire encoding at the tail (Section 8.2). The range check certifies every segment at the balanced alphabet $\{-b^\star/2, \ldots, b^\star/2 - 1\}$ of that level's certified range base $b^\star$, and admissibility enforces $b_1 \leq b^\star$ for completeness (Section 9.3). Extraction may assume only the certified bound, so the $\mathbf{B}$ and $\mathbf{D}$ collision norms, and the inventory, use $b^\star - 1$; the value $b_1 - 1$ would *under*estimate the adversary whenever $b_1 < b^\star$.

The same rule applies elsewhere. Every compression map has collision norm one, since the fused constraint $w(w + 1) = 0$ on its complete digit span enforces $\{-1, 0\}$ (Remark 6.2), and the folded response planes are bounded by the recomposition (77) of the certified alphabet, not by the norm of the honest response. A collision norm is thus a property of the level that scans a segment, not of the level that commits it.

For a source commitment opened under several scanning profiles, $b^\star - 1$ is only the within-occurrence diameter; its $\mathbf{B}$ instance uses the maximum cross-profile diameter of (167). The opening-local $\mathbf{D}$ path uses its scanning level's diameter, and $\mathbf{F}$ and $\mathbf{H}$ keep diameter one.

Uniqueness across opening profiles. Fix a schedule. For a source-group occurrence $\nu = (h, g)$, write its source commitment and the ordered projection of its fold input as

$$\varphi(\nu) := \mathsf{cm}_{h,g} = (\mathsf{shape}_{\mathsf{com},g}, \mathbf{u}_{\mathsf{pub},g}), \qquad \varphi_h := (\varphi(h, g))_{g \in [G_h]}, \tag{178}$$

and write $\mathsf{Src}_\nu(\omega)$ for the component of $\mathsf{Src}(\omega)$ belonging to occurrence ν.

Corollary 10.25 (Source-extractor uniqueness across opening profiles). *Fix a schedule. Let ω, ω' be extracted weak openings containing occurrences ν, ν', possibly at different points and with different claimed values and opening-side $\mathbf{D}/\mathbf{H}$ payloads, such that ν, ν' lie in the same source-comparison class C and $\varphi(\nu) = \varphi(\nu')$. If $\mathsf{Src}_\nu(\omega) \neq \mathsf{Src}_{\nu'}(\omega')$, one can deterministically output a short nonzero kernel vector for the corresponding $\mathbf{A}$, $\mathbf{B}$, or $\mathbf{F}$ view. The $\mathbf{A}$ collision has p_C-norm at most $\eta_{A,C}$; a source-path collision is bounded by the cross-profile certified-alphabet diameter of C. Consequently, two full fold extractions with the same ordered projection φ_h have equal source projections componentwise, or yield one of these collisions.*

The same conclusion holds when one side is a canonical setup reference occurrence $\nu^0_{S,h}$, with $\varphi(\nu^0_{S,h})$ equal to its registered commitment. That side uses the canonical algebraic decommitment and the certificate (162) and needs no evaluation point or opening-side data. Its collision bounds are the mixed envelopes with the reference profile (163).

Proof. Apply Lemma 4.3 to the two openings of the common $\mathbf{B}/\mathbf{F}$ payload at its scheduled depth. Without compression, a disagreement gives a $\mathbf{B}$ collision directly. With compression, a difference in chain digits gives a $\mathbf{B}$ or $\mathbf{F}$ collision with norm at most the cross-profile alphabet diameter of the comparison class. Otherwise the inner-image digits $\hat{\mathbf{t}}$ agree. If a source block still differs, clear denominators with its two weak folding factors as in the proof of Lemma 10.23; (164)–(166) give its scheduled norm bound. Neither argument uses the opening point or the $\mathbf{D}/\mathbf{H}$ path.

For a canonical setup reference, the same proof uses $\bar{c}' = 1$ and $\mathbf{r}' = \mathbf{s}^0$. Its canonical commitment state satisfies the source-path equations and digit alphabets by construction, and (163) supplies the second occurrence profile in the mixed-envelope calculation.

On a Euclidean route, the enforced radii bound the $\mathbf{A}$ collision directly.

Corollary 10.26 (Radius-to-collision bookkeeping). *Suppose each folding challenge c satisfies $\|c\|_1 \le \kappa_1$, $\|c\|_{\mathsf{mul},2} \le \Gamma$, and $\|c\|_2 \le \kappa_2$, and each accepted folded response $\mathbf{z}$ satisfies $\|\mathbf{z}\|_2 \le Z_2$ and $\|\mathbf{z}\|_{\mathsf{mul},2} \le Z_{\mathsf{mul}}$. Then the Euclidean $\mathbf{A}$-collision extracted from two weak openings obeys*

$$\|\mathbf{z}_A\|_{2,\mathrm{coef}}^2 \le 64 \min\left\{\kappa_1^2 Z_2^2, \Gamma^2 Z_2^2, \kappa_2^2 Z_{\mathsf{mul}}^2\right\}. \tag{179}$$

Proof. Take centered integer lifts of the challenges and accepted responses, and form their differences over $\mathbb{Z}$ without reducing them modulo q. By the triangle inequality, the challenge differences have norms at most $2\kappa_1$, 2Γ, and $2\kappa_2$, and the response differences have norms at most $2Z_2$ and $2Z_{\mathsf{mul}}$. Clearing the two unit denominators gives an integer representative of the collision as a difference of two negacyclic products. Applying Lemma 3.1 to these unreduced products gives

$$\|\mathbf{z}_A\|_2 \le 8 \min\{\kappa_1 Z_2, \Gamma Z_2, \kappa_2 Z_{\mathsf{mul}}\}.$$

Only then reduce the collision coefficient-wise to its centered representative modulo q, which cannot increase its Euclidean norm. Squaring gives (179). The argument does not assume that coefficient-wise reduction contracts a convolution operator norm and requires no no-wrap premise.

The radii Z_2 and Z_{mul} are measured on centered ring coefficients. A norm certified before the extension-field packing map ψ is converted by the logical-to-ring factor of Section 3.1 exactly once; the factor is not applied to an already packed $\mathbf{z}$ or to a $\mathbf{z}$ reconstructed from base-field digit planes.

Incoming tensor-reduction data. Let h be an evaluation-trace receiver. It begins with the shared tensor reduction of Section 5.2 on the ordered multi-instance interface of Section 5.1; the next definition and lemma extract its incoming claims. A nonterminal receiver has $h \ge 2$; the terminal may occur earlier when a shorter schedule reaches $\mathsf{TerminalInnerState}$. Write $\mathcal{G}_h^{\mathsf{in}}$ for its ordered incoming groups, $I_{h,g}$ for the claims in group g, and $P_{h,g}$ for that group's binding payload: the ordinary outer commitment at a nonterminal receiver, and the predecessor-bound canonical inner t state at the terminal. Claim c has Boolean evaluation vector $f_{h,g,c}$, native point $\mathbf{r}_{h,g}$, and value $v_{h,g,c}$, and its input assertion is

$$\widetilde{f_{h,g,c}}(\mathbf{r}_{h,g}) = v_{h,g,c}.$$

Definition 10.27 (Incoming claims for tensor reduction). *For the incoming data above, set $N_{\mathsf{clm},h} := \sum_g |I_{h,g}|$. The receiver uses the following rules.*

(i) *The schedule fixes every group descriptor, basis, packing bijection $\mathsf{Pack}_{h,g}$, and tail arity $m_{h,g}$; let $m_{\max,h} := \max_g m_{h,g}$. The packed multilinear polynomial*

$$\bar{f}_{h,g,c} := \mathsf{Pack}_{h,g}(f_{h,g,c})$$

is obtained by canonically regrouping the entries of $f_{h,g,c}$ into extension-field Boolean evaluations and interpolating them, as in Section 3.7. This packing map is invertible on the Boolean evaluations.

(ii) *The transcript binds these descriptors together with the incoming payloads, native points, and values before the receiver begins its shared tensor-reduction prefix. Each group's input point remains part of its bound statement.*

(iii) *On a nonidentity route, the receiver applies the canonical shared reduction of Theorem 5.1. For its final point $\boldsymbol{\rho}_h$, group g uses the native prefix $\boldsymbol{\rho}_{h,g} := \boldsymbol{\rho}_{h,[m_{h,g}]}$ and*

$$\theta_{h,g} = A_{\eta_h,g}(\boldsymbol{\rho}_{h,g}) \prod_{t=m_{h,g}}^{m_{\max,h}-1} (1-\rho_{h,t}), \qquad \tilde{v}_{h,g,c} = \theta_{h,g} \bar{f}_{h,g,c}(\boldsymbol{\rho}_{h,g}).$$

The verifier rejects if any $\theta_{h,g}$ is zero.

(iv) *On the identity route, the prefix and its challenges are omitted,* $\theta_{h,g} = 1$*, and the canonical packing identification forwards each claim.*

(v) *After binding the shared evaluation-trace opening payload, the receiver sets the first coefficient to one and samples the remaining* $N_{\mathsf{clm},h} - 1$ *entries of a fresh normalized coefficient vector for the final products.*

Extraction error. The tensor-reduction batch has coordinate-wise extraction charge

$$\varepsilon_{\mathsf{tensor},h} := \begin{cases} 0, & \text{identity tensor reduction,} \\ \dfrac{\kappa_h + (N_{\mathsf{clm},h} - 1) + 2m_{\max,h}}{|E_h|}, & \text{otherwise,} \end{cases}$$

as in (282), and claim-batching tree factor $N_{\mathsf{clm},h}$; its one-shot bound replaces $N_{\mathsf{clm},h} - 1$ by $\mathbf{1}_{\{N_{\mathsf{clm},h}>1\}}$. The subsequent post-payload claim batch has coordinate-wise charge

$$\varepsilon_{\mathsf{grp},h} := \frac{N_{\mathsf{clm},h} - 1}{|E_h|}, \tag{180}$$

tree factor $N_{\mathsf{clm},h}$, and one-shot failure at most $\mathbf{1}_{\{N_{\mathsf{clm},h}>1\}}/|E_h|$.

Lemma 10.28 (Tensor reduction in the receiving fold). *Fix incoming claims as in Definition 10.27. Suppose the extractor has*

(i) *the complete mixed accepting tree for the shared tensor-reduction prefix: a nested binary tree in* η*, a flat affine claim-batching family, and scalar sum-check subtrees; and*

(ii) *below every leaf of this prefix, a descendant-certified evaluation-trace source whose canonical base-coordinate unpacking authenticates each fixed payload* $P_{h,g}$ *under the scheduled source relation and satisfies every forwarded packed claim.*

Then a deterministic extractor returns, for every group g *and claim* $c \in I_{h,g}$*, a Boolean evaluation vector* $f_{h,g,c}$ *authenticated by* $P_{h,g}$ *with* $\widetilde{f_{h,g,c}}(\mathbf{r}_{h,g}) = v_{h,g,c}$*; otherwise it returns a commitment-matrix collision from Corollary 10.25 at a nonterminal receiver or a terminal* $\mathbf{A}$*-matrix collision. The required receiver tree incurs coordinate-wise error* $\varepsilon_{\mathsf{tensor},h} + \varepsilon_{\mathsf{grp},h}$*; an ordinary accepting transcript instead incurs the two one-shot errors above.*

Proof. Tensor prefix. Apply Theorem 5.1 to the fixed payloads, points, values, shapes, and packing bases. First certify every descendant and use source binding to identify its packed polynomial. Backward scalar interpolation certifies the degree-2 sum-check for every tensor-batch child. The flat normalized claim family then separates the claim discrepancies for each fixed η, and Lemma 10.13 applied to the nested binary η tree forces every tensor-row discrepancy to vanish identically.

Unpacking. Inverting each fixed packing map gives the Boolean evaluation vectors $f_{h,g,c}$. If two descendants unpack to different evaluation vectors under the same $P_{h,g}$, source binding gives the stated matrix collision; at the terminal, subtracting their predecessor-bound inner-state equations gives the $\mathbf{A}$ collision. The identity route gives the same vectors directly.

Final batch. After the shared opening payload is bound, the fresh evaluation-trace claim batch separates the final products with the charge (180). Cylindrical extension changes only the sum-check domain, so the incoming claims remain separately bound. Later fold challenges may differ across tensor-reduction siblings: the extractor first certifies the complete descendant of each sibling and never assumes equality of a challenge sampled after the tensor-reduction node.

Use along the schedule. The first evaluation-trace receiver composes the predecessor extractor with its own tensor-reduction prefix. A nonterminal receiver then runs its fold and produces an ordinary opening; the terminal instead binds the predecessor state. When setup offloading supplies a recursive-witness group and a setup group, both enter the same shared tensor reduction. The incoming claims remain separately bound, and each output claim uses its prescribed prefix of the new sum-check point.

Terminal statement and transcript order. Every admissible schedule ends in the evaluation-trace terminal of Section 8.2, which has exactly one incoming recursive-witness group and no pending setup opening. Write $R_{\mathsf{term}} = \mathbb{Z}_q[X]/(X^{d_{A,\mathsf{term}}}+1)$ and let B_{term} be its number of source blocks. We analyze the terminal that sends every block; Lemma 10.47 then transfers the result to the variant of Section 8.2, which replaces the last $\mathbf{t}$ block and the last partial by digests. The terminal's tensor reduction and direct checks follow Section 8.2, in this transcript order:

(1) before any terminal challenge, the predecessor binds the canonical state $(\mathbf{t}_i)_{i\in[B_{\mathsf{term}}]}$ with $\mathbf{t}_i = \mathbf{A}_{\mathsf{term}}\mathbf{s}_i$, and the terminal takes that same state as public input;
(2) the tensor reduction fixes one packed claim and $v_{\mathsf{tensor}} = \theta_{\mathsf{term}}\bar{v}_{\mathsf{term}}$, with $\theta_{\mathsf{term}} = 1$ on the identity route and $\theta_{\mathsf{term}} \neq 0$ otherwise;
(3) the terminal fixes the clear partials $(e_i)_i$ and only then samples $(c_i)_i$;
(4) it absorbs $\mathbf{z} = \sum_i c_i\mathbf{s}_i$ and verifies (128)–(130) together with the scheduled clear encoding and norm bounds.

These are direct checks; the terminal has no outer $\mathbf{u}$, $\mathbf{B}$, $\mathbf{D}$, quotient, or relation sum-check.

Terminal collision bounds. Let $\kappa_{1,\mathsf{term}}$ bound the terminal challenge ℓ_1 norm and let $Z_{\infty,\mathsf{term}}$ bound the accepted clear response coefficient norm. The terminal source occurrence therefore has $K_{1,\mathsf{term}} = 2\kappa_{1,\mathsf{term}}$ and $B_{\infty,\mathsf{term}} = 2Z_{\infty,\mathsf{term}}$, and its diagonal comparison-class radius on the coefficient route is

$$\eta_{A,\mathsf{term},\infty} = 8\kappa_{1,\mathsf{term}}Z_{\infty,\mathsf{term}}. \tag{181}$$

On a Euclidean route, let Γ_{term} be the enforced bound on the challenge operator norm and let $S_{\max,\mathsf{term}}$ be the enforced response squared-norm bound. Then

$$K_{\mathsf{mul},2,\mathsf{term}} = 2\Gamma_{\mathsf{term}}, \quad B_{2,\mathsf{term}} = 2\sqrt{S_{\max,\mathsf{term}}}, \quad \eta^2_{A,\mathsf{term},2} = 64\Gamma^2_{\mathsf{term}}S_{\max,\mathsf{term}}. \tag{182}$$

The terminal $\mathbf{A}$ rank is admissible only if it meets the security floor for the selected one of these two instances.

Proposition 10.29 (Akita terminal extraction certificate). *Consider the terminal and bounds above. From the terminal's mixed accepting tree, with a nested tensor-prefix tree and coordinate-wise fold siblings, a deterministic extractor returns a descendant certificate for its incoming openings or a short $\mathbf{A}$-matrix collision. The exact terminal equations add no field error. Its fold challenges contribute*

$$\varepsilon_{\mathsf{term}} := \frac{B_{\mathsf{term}}}{|\mathcal{C}^{\mathsf{acc}}_{\mathsf{term}}|}, \tag{183}$$

while the terminal tensor-reduction term is charged separately. There is no terminal claim-batching term because the terminal has one claim.

Proof. Source blocks. The predecessor-bound $\mathbf{t}$ state, the tensor-reduction output, and the revealed partials $(e_i)_i$ are fixed before the fold challenges. For each i, the CWSS structure supplies two children with challenges $\mathbf{c}^{(i,0)}$ and $\mathbf{c}^{(i,1)}$ that agree outside coordinate i and differ at i. With their responses $\mathbf{z}^{(i,0)}$ and $\mathbf{z}^{(i,1)}$, set

$$\Delta c_i := c_i^{(i,1)} - c_i^{(i,0)}, \qquad \mathbf{s}_i^\star := (\Delta c_i)^{-1}(\mathbf{z}^{(i,1)} - \mathbf{z}^{(i,0)}).$$

The accepted challenge family makes Δc_i a unit in R_{term}. Subtracting the two $\mathbf{A}$ equations and cancelling Δc_i gives $\mathbf{A}_{\mathsf{term}}\mathbf{s}_i^\star = \mathbf{t}_i$, and subtracting the two evaluation-trace fold-evaluation equations gives

$$e_i = \mathbf{a}_{\mathsf{term}}^\top \mathbf{G}_{b_{\mathsf{term}},M_{\mathsf{term}}}\mathbf{s}_i^\star \quad \text{in } R_{\mathsf{term}}.$$

Repeating this for every coordinate extracts the terminal source blocks and their partials. If two extracted candidates for one block disagree, both map to the same $\mathbf{t}_i$, and clearing their unit challenge differences as in Lemma 10.23 gives a short $\mathbf{A}$ collision bounded by (181) or (182).

Trace equation and error. The trace equation has no fold challenge, and the verifier checks it on every accepting child against the fixed pair $(\theta_{\mathsf{term}}, v_{\mathsf{tensor}})$, so the extractor reads it from one child. It connects the extracted partials to the tensor reduction output, and Lemma 10.28 connects that output to the terminal's one incoming base-field opening claim. The revealed segment encoding lets the extractor check all digit and norm predicates directly. Coordinate-wise CWSS gives (183); every remaining terminal check is an exact equality.

One-fold extraction. The fold extractor uses transcripts that branch at the folding challenge.

Definition 10.30 (Nested coordinate-wise fold family). *Fix a nonterminal fold with G source groups and let*

$$\mathcal{Q} := \{(g, c, i) : g \in [G],\ c \in [n_{\mathsf{clm},g}],\ i \in [B_g]\}$$

index its folding challenges. A nested coordinate-wise fold family *consists of:*

(i) *one transcript prefix ending immediately before this fold challenge;*

(ii) *$|\mathcal{Q}|+1$ folding-challenge children forming a coordinate-wise 2-special-sound structure: for every $q \in \mathcal{Q}$, a pair $\mathbf{c}^{(q,0)}, \mathbf{c}^{(q,1)}$ of these children agrees outside coordinate q and differs at q by a unit with the norm prescribed by its source instance (the pairs need not share a common child); and*

(iii) *below every folding-challenge child, a complete accepting mixed subtree, as specified by Lemma 10.14, for the public-coin rounds that follow that challenge: the scheduled ring-reduction, range, norm, and fused-relation rounds, followed by the recursive protocol that authenticates the outgoing opening claim.*

The subtrees below two different folding children need not use the same later challenges or contain the same later prover messages. The family is descendant-certified *if the recursive extractor has already authenticated every outgoing opening claim at the leaves of these strong-check subtrees, and, for each fixed folding child, all such openings under the same outgoing commitment yield the same commitment-side source vector.*

If two such openings yield different source vectors, Corollary 10.25 already returns a matrix collision.

Theorem 10.31 (One-fold backward extraction). *Fix an extraction-admissible schedule and one of its nonterminal folds. Given a descendant-certified nested coordinate-wise fold family (Definition 10.30), a deterministic extractor returns either:*

(i) *a weak opening of every incoming source group, satisfying the exact commitment, partial-evaluation, and public opening equations of Definition 10.22; or*

(ii) *a short nonzero kernel vector for one matrix view in $\mathcal{I}(\mathsf{sch})$ at the current level.*

For source instance g, the extracted coefficient-difference bound is $\bar{\beta}_{\infty,g} = \Delta^{\mathsf{cert}}_{f,g}$. If the schedule enforces a Euclidean radius profile from Corollary 10.26, the weak opening also satisfies the certified Euclidean difference bounds. The extractor runs in time polynomial in the supplied transcript tree and does not require challenges sampled after the folding vector to agree across two folding children.

Moreover, for two successful invocations whose input instances have the same source projection φ_h, either their outputs have the same Src *projection or Corollary 10.25 returns a current-level matrix collision. Thus the extracted source is unique relative to φ_h, although the complete point-dependent weak openings need not be.*

Proof. Exact branch relations. Fix one folding-challenge child. Starting at the authenticated outgoing claims, run the strong extractors from the leaves of its subtree toward the folding node. The sum-check extractors force the range, norm, and fused relation identities, and the scheduled quotient-lift or quotient-free extractor then recovers the original mixed-dimension ring identities. Thus the child yields one exact instance of every row of the schedule's mode-selected relation (the native carrier relation at levels 0 and 1, or the established evaluation-trace relation at later nonterminal levels), together with the certified digit and response bounds. The norm claims are sent inside Stage 1, so they are fixed only within the subtree below the range challenges that precede them and may differ between its branches. No equality of them across branches is needed: every branch reads the same authenticated successor table, and one accepting branch gives its integer response bound. Repeat this independently below every folding-challenge child. Each recovered identity holds over its native ring, so no later challenge appears in it.

Digit paths. Compare every commitment path and the shared opening path across these exact branch relations, using the scheduled depth in Lemma 4.3. A disagreement on an uncompressed path gives a **B** or **D** collision directly; on a compressed path it gives a collision in its source matrix or one of its chain matrices. Assume from now on that the source and chain digits agree across all branches.

Coordinate siblings. Fix $q = (g, c, i) \in \mathcal{Q}$ and subtract the exact relation for child $(q, 0)$ from that for child $(q, 1)$. Every other folding term cancels inside instance g. With $\bar{c}_{g,c,i} := c^{(q,1)}_{g,c,i} - c^{(q,0)}_{g,c,i}$, the response rows give

$$\iota_g(\bar{c}_{g,c,i})\mathbf{s}_{g,c,i} = \mathbf{z}^{(q,1)}_g - \mathbf{z}^{(q,0)}_g,$$

which the certified response digits bound by $\Delta^{\mathsf{cert}}_{f,g}$ and which satisfies any enforced Euclidean bounds. Subtracting the two fold-consistency and fold-evaluation rows, then cancelling the challenge difference in the ring of each row, gives

$$\mathbf{A}_g \mathbf{s}_{g,c,i} = \mathbf{G}_{b_{1,g},n_{A,g}} \hat{\mathbf{t}}_{g,c,i}, \qquad e_{g,c,i} = \mathcal{L}_g(\mathbf{G}_{b_g,M_g} \mathbf{s}_{g,c,i}).$$

The first inversion is in the ambient ring through $\iota_g(\bar{c}_{g,c,i})$, and the second is in the scheduled opening ring S_g (and its scalar extension when needed). With coefficient packing, Lemma 4.2 supplies both units from the same carrier difference. With evaluation trace, S_g is the ambient folding ring and ι_g is the identity, so the schedule's unit-difference condition supplies both inversions directly.

Assembly. Repeating this for every q extracts all semantic blocks. The common digits already satisfy their compression chains and scheduled public opening row, because each extracted branch satisfies every row exactly, so they form the claimed weak opening. The final uniqueness statement is Corollary 10.25 applied to the two extracted weak openings.

The theorem is thus one step of the backward induction in Theorem 10.40, not a standalone knowledge-soundness claim for an unauthenticated outgoing opening.

The exact branch relation. Only the first step of the proof above reads the subtree below a folding-challenge child. The Fiat–Shamir extractor replaces that step by a test (Lemma 10.42 and Proposition 10.44), so we isolate what it delivers. Fix a nonterminal fold and a transcript prefix ξ through the messages of step 3 of Figure 4, ending before α; thus ξ contains the fold challenges and the successor payload. A table $\mathbf{w}$ on $\{0,1\}^{\mu'}$, read in the witness order of (76), *satisfies the exact branch relation of* ξ if:

(i) every entry lies in the alphabet certified for its witness segment, and in $\{-1,0\}$ on $\mathcal{I}_{\mathsf{bin}}$;
(ii) every row of the schedule's mode-selected relation (Figure 5) holds as an identity in its native ring, for the commitments, opening payload, scalar row, and fold challenges in ξ; and
(iii) on a Euclidean route, the response digits in $\mathbf{w}$ satisfy $\mathcal{E}_{\mathsf{int}}(\hat{\mathbf{z}}) \le S_{\max}$.

The Fiat–Shamir proof uses three properties of this relation.

(a) It can be decided in polynomial time from ξ and $\mathbf{w}$, so an extractor can test it.
(b) It refers to no later challenge and to no short opening of the successor payload, so it is a property of the pair $(\xi, \mathbf{w})$ alone. Conditions (ii) and (iii) do not depend on the quotient entries of $\mathbf{w}$, which enter only through their alphabet in (i).
(c) It refers to no norm message. The norm claims are sent later, in Stage 1, and the verifier's checks on them (canonical decoding, the integer reconstruction (88), and the comparison $\mathcal{E}_{\mathsf{resp}} \le S_{\max}$) together with the norm identities certify condition (iii) (Lemma 10.42).

Corollary 10.32 (One-fold extraction from exact branch relations). *Fix one nonterminal fold of an extraction-admissible schedule and a transcript prefix ending immediately before its fold challenge. Consider $|\mathcal{Q}|+1$ extensions of this prefix through step 3 whose fold challenges form the coordinate-wise structure of Definition 10.30. Suppose that each extension passes every verifier check that it determines and comes with a table that satisfies its exact branch relation. Then a deterministic algorithm computes from these extensions and tables one of the two alternatives of Theorem 10.31, with the bounds stated there, in time polynomial in the explicit instance-and-witness length. It uses no transcript below a folding-challenge child.*

Proof. Every step of the proof of Theorem 10.31 after the first uses only the common prefix, the exact branch relation of each child, and its certified digit and response bounds. Condition (i) supplies the digit bounds. On a Euclidean route, condition (iii) supplies the response bound.

Strong Checks and Error Accounting The results below prove sound, one at a time, public batching and the checks after the fold challenge, from which Theorem 10.31 recovers the exact branch relation below each folding child. The paragraphs after them explain how these strong checks nest inside the weak fold and add all errors into the schedule error $\varepsilon_{\mathsf{int}}(\mathsf{sch})$ of (194), the knowledge error of Theorem 10.40 and, multiplied by $Q+1$, the leading term of Theorem 10.46.

Lemma 10.33 (Per-claim soundness of public batching). *Consider the coefficient-packing application root prescribed by (127). Let the points, the claimed values, and values $\bar{v}_\ell$ for $\ell \in [\Lambda]$ be fixed before the verifier draws ζ_{batch}. In the extraction proofs, $\bar{v}_\ell$ is the padded value of the extracted decoded vector for global claim ℓ, which the source candidate determines. If $\bar{v}_\ell \neq \phi_{g(\ell),c(\ell)} v_{g(\ell),c(\ell)}$ for some $\ell \in [\Lambda]$, then*

$$\sum_{\ell \in [\Lambda]} \zeta_{\mathsf{batch}}^{\ell} \bar{v}_\ell = \sum_{\ell \in [\Lambda]} \zeta_{\mathsf{batch}}^{\ell} \phi_{g(\ell),c(\ell)} v_{g(\ell),c(\ell)}$$

holds with probability at most $(\Lambda - 1)/|E|$.

Proof. The difference is a nonzero polynomial in ζ_{batch} of degree at most $\Lambda - 1$; apply Schwartz–Zippel.

Lemma 10.34 (Soundness of the Euclidean response check). *Fix a nonterminal Euclidean route from Section 6.2. Let N_{pair} be the number of digit-inner-product claims and δ_f the response digit depth. Apart from the ordinary sum-check extraction errors, an accepting range check, Stage 1 norm check, Stage 2 binding to the response digits, and integer reconstruction imply*

$$\mathcal{E}_{\mathsf{int}}(\hat{\mathbf{z}}) = \mathcal{E}_{\mathsf{resp}} \leq S_{\max}, \qquad \|\mathbf{z}\|_{2,\mathrm{coef}}^2 \leq \mathcal{E}_{\mathsf{resp}}. \tag{184}$$

The additional random-linear-combination error is at most

$$\varepsilon_{\mathsf{norm}}^{\mathsf{rlc}} = \begin{cases} 2|E|^{-1}, & \text{direct norm check,} \\ (N_{\mathsf{pair}} + \delta_f)|E|^{-1}, & \text{digit-expanded norm reconstruction.} \end{cases} \tag{185}$$

At the Akita terminal, the verifier obtains (184) by decoding the response and computing the integer square sum directly, with no sum-check or random-linear-combination error.

Proof. The norm claims may depend on every challenge drawn before them. Each batching bound below is conditional on the transcript before its own challenge, which is drawn after the claims it combines, so this dependence does not change the bounds. The integer equality follows from Lemma 6.3 on the direct route and from Lemma 6.4 and Proposition 6.5 on the digit-expanded route. Injectivity of the response-digit map and monotonicity under centered reduction give the second inequality in (184).

The range/norm merge contributes at most $1/|E|$. The Stage 2 binding to the response digits has degree one on the direct route and at most δ_f on the digit-expanded route, contributing at most $1/|E|$ and $\delta_f/|E|$, respectively. Batching the N_{pair} inner-product claims on the digit-expanded route adds $(N_{\mathsf{pair}}-1)/|E|$. Summing gives (185).

Corollary 10.35 (Route-specific inner-commitment collision). *Fix one source instance and its scheduled $\mathbf{A}$ matrix.*

(i) Coefficient route. *Suppose every challenge satisfies $\|c\|_1 \leq \kappa_1$ and every accepted response satisfies $\|\mathbf{z}\|_{\infty,\mathrm{coef}} \leq Z_\infty$. Then the extracted coefficient-norm collision has radius $p_A = \infty$, $\eta_{A,\infty} = 8\kappa_1 Z_\infty$.*

(ii) Euclidean route. *Suppose every accepted challenge has multiplication-operator norm at most Γ, the recursive check or terminal enforces the bound $S_{\max}$. Then the extracted collision has $p_A = 2$, $\eta_{A,2}^2 = 64\Gamma^2 S_{\max}$.*

For a coefficient-carrier source, the certified carrier bound is also the operator bound of the embedded challenge on the ambient response by Lemma 4.2; no lane or extension-degree factor is added. A schedule may use the Euclidean radius only if it certifies the accepted challenge family and its operator bound, the response-norm check and bound, and a Module-SIS table entry for the resulting radius. Its Module-SIS inventory may use the exact radius or an upward-rounded one.

Proof. Taking two accepting openings doubles the challenge and response radii. The two terms that clear their weak-opening denominators supply the remaining factor of two. The coefficient statement follows from (176). The Euclidean statement follows from Lemma 10.34 and Corollary 10.26. An upward-rounded radius still contains every collision produced by the reduction.

For source instance g, a challenge family drawn from an ℓ_1-ball of mass ω_g has $\bar{\kappa}_{1,g} \leq 2\omega_g$, so the coefficient-norm $\mathbf{A}$ instance of that source has radius $\eta_{A,g} = 4\omega_g \Delta_{f,g}^{\mathsf{cert}}$. On the Euclidean route, a schedule that certifies an exact operator bound on challenge differences may use it in place of 2Γ in (177); the corollary bounds the two challenges separately.

Theorem 10.36 (Ring reduction and sum-check CWSS). *For a nonterminal fold, let $d_{\max}$ be the largest native ring dimension among its active ring-valued rows. Its ring-reduction coordinate is k_{ring}-special sound, where*

$$k_{\mathsf{ring}} := \begin{cases} 2d_{\max}, & \mathsf{RingCheck} = \mathsf{QuotientLift}, \\ d_{\max}, & \mathsf{RingCheck} = \mathsf{QuotientFree}. \end{cases} \tag{186}$$

The fused relation sum-check has μ' rounds of degree-3 messages.

Proof. The lifted residuals have degree below $2d_{\max}$ by Lemma 7.1; the reduced residuals have degree below $d_{\max}$ by Theorem 7.4. Interpolation gives the two arities. For each fixed ring-reduction challenge, the relation-row discrepancy is multilinear in τ_1. Its nested binary interpolation tree has $2^{\lceil \log_2 n_{\mathsf{rel}} \rceil}$ leaves and forces all rows by Lemma 10.13; coordinate pairs alone would not suffice. The final claim follows from the degree-3 summand in (125) and Theorem 3.3.

Lemma 10.37 (CWSS of the digit range tree). *The digit range tree of Section 6.1 is special sound round by round. If it has stages of inner degrees $d_1, \ldots, d_S$ and interstage frontiers of sizes $m_1, \ldots, m_{S-1}$, the digit range check contributes*

$$\varepsilon_{\mathsf{tree}} = \sum_{j=1}^{S} \frac{\mu'(d_j + 1)}{|E|} + \sum_{j=1}^{S-1} \frac{m_j - 1}{|E|} \tag{187}$$

to the interactive knowledge error, where μ' is the extended-witness variable count. Moreover:

(i) *The range point τ_0 uses a nested binary tree of depth μ'. Once its descendants certify the sum in (80), Lemma 10.13 forces every Boolean range residual to be zero. Its charge $\mu'/|E|$ is listed separately in (189).*

(ii) *The binary-support residual is multilinear in $\mathbf{r}_{\mathsf{virt}}$, a point already supplied by the nested scalar challenge tree of the last range sum-check. After the descendants of this point certify the fused relation and separate ζ_{bin}, two children per coordinate of that tree suffice for binary-support interpolation, so this check adds no new tree factor.*

Remark 10.38 (Opening-row binding). The extracted partials satisfy the field-level opening row already in the relation: (100) with coefficient packing, or (101) with evaluation trace. The same τ_1 batches this row with the commitment, fold, and compression rows, so its row-batching and relation-check errors are already in (189) and it has no separate term. The preceding tensor reduction is a separate strong reduction, charged by (191).

The nested strong–weak invariant. SuperNeo formalizes two interfaces for this lattice-extraction problem [84, Defs. 9–10, Thm. 8]. A weak interactive reduction extracts a witness for a relaxed relation, and repeated extractions with the same public projection produce the same witness. A strong interactive reduction recovers its input witness when its output witness is unique. Their theorem composes a strong reduction followed by a weak one. Akita uses the same invariant: the relaxed relation is Definition 10.22, the public projection is φ_h from (178), and the witness subject to uniqueness is $\mathsf{Src}(\omega)$, not the complete weak opening, whose partial opening evaluations and $\mathbf{D}/\mathbf{H}$ paths may legitimately change with the opening point.

Akita does not use the sequential theorem as a black box, because its weak fold and strong checks are nested. A folding challenge precedes the ring-reduction, range, norm, and relation challenges, and its outgoing claim is authenticated only by the recursive suffix. The outgoing commitment is formed after the fold challenge and may differ across folding children, so it does not provide the common intermediate projection that SuperNeo's strong-reduction definition requires at that cut.

The composition takes two forms.

– *Complete tree.* Below each folding child, the extractor recovers the exact native-ring branch relation by interpolation over the strong checks, and then compares coordinate siblings at the folding node. This is the one-fold step of Theorem 10.31, which Theorem 10.40 iterates backward from the terminal.

– *Checkpoints.* Once the outgoing source is authenticated, the exact branch relation is a decidable property of it, and the checks between the outgoing commitment and the next fold challenge are public-coin tests of it. The Fiat–Shamir extractor tests the relation instead of interpolating, as in the paragraph on extraction with checkpoints (Proposition 10.44), and needs no transcript tree for the strong checks.

In both forms, every failure of extraction uniqueness is a matrix-view kernel vector, which Lemma 10.20 converts to ordinary MSIS.

Per-level knowledge error. Write $\mathsf{Lev}_{\mathsf{nt}}(\mathsf{sch})$ for the nonterminal fold levels of a schedule sch. At level h:
- $\mathcal{G}_h$ lists the source groups in order. Group $g \in \mathcal{G}_h$ has $W_{h,g}$ fold-challenge coordinates (its number of claims times its number of source blocks), and $\mathcal{C}^{\mathsf{acc}}_{h,g}$ is the exact family the verifier samples after every witness-independent filter;
- μ'_h is the extended-witness variable count, n_h the number of rows of the extended relation matrix, and $k_{\mathsf{ring},h}$ the ring-reduction arity from (186);
- $\beta_h \in \{0,1\}$ indicates that the schedule-derived binary support is nonempty;
- the range tree has stage degrees $d_{h,1}, \dots, d_{h,S_h}$ and interstage frontier sizes $m_{h,1}, \dots, m_{h,S_h-1}$; for every Euclidean source group, its last stage includes the fused norm term of (90).

The fold errors are

$$\varepsilon_{\mathsf{cw},h} := \sum_{g \in \mathcal{G}_h} \frac{W_{h,g}}{|\mathcal{C}^{\mathsf{acc}}_{h,g}|}, \qquad \varepsilon_{\mathsf{cw}}(\mathsf{sch}) := \sum_{h \in \mathsf{Lev}_{\mathsf{nt}}(\mathsf{sch})} \varepsilon_{\mathsf{cw},h} + \varepsilon_{\mathsf{term}}. \tag{188}$$

For one ordinary opening, $W_{h,g} = B_h$ and $\varepsilon_{\mathsf{cw},h} = B_h/|\mathcal{C}^{\mathsf{acc}}_h|$. With an operator-norm filter, the denominator uses the accepted family (23), which response grinding does not change. The norm-claim numerators are $N^{\mathsf{norm}}_h := \sum_{g \in \mathcal{G}_h} N^{\mathsf{norm}}_{h,g}$, where

$$N^{\mathsf{norm}}_{h,g} := \begin{cases} 0, & \text{coefficient-}\ell_\infty \text{ route}, \\ 2, & \text{direct Euclidean route}, \\ N_{\mathsf{pair},h,g} + \delta_{f,h,g}, & \text{digit-expanded Euclidean route}. \end{cases}$$

The interactive knowledge error of one scheduled nonterminal level is

$$\begin{aligned} \varepsilon_h = \varepsilon_{\mathsf{cw},h} + \frac{1}{|E|}\Big(& k_{\mathsf{ring},h} + \lceil \log n_h \rceil + \mu'_h + \beta_h \mu'_h + \sum_{s=1}^{S_h} \mu'_h (d_{h,s} + 1) \\ & + \sum_{s=1}^{S_h - 1} (m_{h,s} - 1) + 1 + \beta_h + N^{\mathsf{norm}}_h + 3\mu'_h \Big). \end{aligned} \tag{189}$$

In order, the terms account for:
- $\varepsilon_{\mathsf{cw},h}$: the fold challenge;
- $k_{\mathsf{ring},h}$: ring reduction (Theorem 10.36);
- $\lceil \log n_h \rceil$: row batching by τ_1;
- μ'_h and $\beta_h \mu'_h$: the range point τ_0 and the binary-support point (Lemma 10.37);
- the two sums: the range-tree stages and frontiers, as in (187);
- 1 and β_h: relation/range batching and binary batching;
- N^{norm}_h: norm-claim binding, the exact total numerator of every Stage 1 norm merge, Stage 2 shifted-power binding to the response digits, and digit-inner-product-claim batch at level h, by (185);
- $3\mu'_h$: the relation check, whose fused sum-check has μ'_h rounds of degree 3.

The terminal has no copy of (189). Its checks in the clear have the shorter layout of Proposition 10.29, with no norm-claim terms (Lemma 10.34), and its fold-CWSS contribution is $\varepsilon_{\mathsf{term}}$ from (183).

Let $o_h \in \{0,1\}$ indicate that nonterminal level h runs the Stage-3 setup product sum-check; $o_h = 1$ for every offloaded edge $h \to h+1$. Let $N^R_{\mathsf{setup},h}$ be the padded setup-domain length of Stage 3 when $o_h = 1$, and 1 otherwise. Adding the degree-2 Stage-3 sum-check gives the schedule-level error

$$\varepsilon^{\mathsf{sch}}_h := \varepsilon_h + o_h \frac{2 \log_2 N^R_{\mathsf{setup},h}}{|E|}. \tag{190}$$

Schedule error. Three schedule-wide terms cover the receivers and application batching. They sum over the following sets.
- $\mathsf{EvalRecv}(\mathsf{sch})$: the evaluation-trace receivers, including the terminal regardless of its absolute depth, with the incoming-group notation of Definition 10.27.

– $\mathsf{Recv}(\mathsf{sch})$: the internal receivers whose mode-specific relation combines $N_{\mathsf{clm},h} > 1$ fixed claims, including a packing receiver with an offloaded setup claim and every evaluation-trace receiver with more than one tensor-reduction output claim. Application-level public batching is excluded.
– $\mathsf{Batch}(\mathsf{sch})$: the public application-batching steps, where step a has Λ_a claims and challenge field E_a.

Define

$$\varepsilon_{\mathsf{tensor}}(\mathsf{sch}) := \sum_{h \in \mathsf{EvalRecv}(\mathsf{sch})} \varepsilon_{\mathsf{tensor},h}, \tag{191}$$

$$\varepsilon_{\mathsf{grp}}(\mathsf{sch}) := \sum_{h \in \mathsf{Recv}(\mathsf{sch})} \frac{N_{\mathsf{clm},h} - 1}{|E_h|}, \tag{192}$$

$$\varepsilon_{\mathsf{batch}}(\mathsf{sch}) := \sum_{a \in \mathsf{Batch}(\mathsf{sch})} \frac{\Lambda_a - 1}{|E_a|}. \tag{193}$$

The normalized coefficients of a receiver are sampled after its commitments, points, and claimed values are fixed at a packing receiver, and after the shared opening payload is fixed at an evaluation-trace receiver. Each summand of (192) is the coordinate-wise extraction charge of its receiver; the ordinary random-linear-combination failure is at most $1/|E_h|$. Application batching remains a separate term because it binds the original public claims rather than an internal recursive boundary.

Lemma 10.39 (Setup offloading inside the scheduled reduction). *Let $h \to h+1$ be an offloaded edge.*

(i) *Its extra error terms are the Stage 3 term in $\varepsilon_h^{\mathsf{sch}}$ and the fresh two-group combination term in $\varepsilon_{\mathsf{grp}}$ of (192). If $h+1 \geq 2$, its two incoming groups also enter the single tensor reduction of the receiver.*
(ii) *Offloading does not change either fold's CWSS extraction or response-norm bounds.*
(iii) *Binding the setup opening of the registered commitment $C_{S,h}$ is covered by the per-matrix Module-SIS inventory $\mathcal{I}(\mathsf{sch})$, at the comparison-class radii of the setup occurrence.*
(iv) *The receiver $h+1$ is nonterminal: the last nonterminal fold must check a pending setup group and may not create another one for the terminal.*

Proof. Stage 2 already produces the recursive-witness opening. Stage 3 is an ordinary degree-2 sum-check over $\log_2 N^R_{\mathsf{setup},h}$ variables, giving its term in (190).

At level $h+1$, the witness and setup prefix are two instances, each at its own point. At an evaluation-trace receiver, Lemma 10.28 first reduces them in one shared tensor reduction, which does not identify distinct incoming opening points. The receiver then samples its normalized coefficient in the order stated after (192); at a packing receiver, this is the scalar of (137). At an evaluation-trace receiver, a nonzero residual vector passes this batch with one-shot probability at most $1/|E|$. In both cases, two distinct values of the normalized coefficient extract both claims, and the mode-selected nonterminal multi-instance fold then handles both groups.

Lemma 9.1 connects an accepted opening of the schedule-selected $C_{S,h}$ to the true setup scan, and the setup opening is checked under the same scheduled matrix inventory as the other opening groups.

Collecting these terms, an upper bound on the interactive knowledge error of the scheduled protocol is

$$\begin{aligned}\varepsilon_{\mathsf{int}}(\mathsf{sch}) := \sum_{h \in \mathsf{Lev}_{\mathsf{nt}}(\mathsf{sch})} \varepsilon_h^{\mathsf{sch}} + \varepsilon_{\mathsf{tensor}}(\mathsf{sch}) + \varepsilon_{\mathsf{grp}}(\mathsf{sch}) \\ + \varepsilon_{\mathsf{term}} + \varepsilon_{\mathsf{batch}}(\mathsf{sch}).\end{aligned} \tag{194}$$

Each public coin is charged once: the first sum covers the nonterminal folds with their strong suffixes and Stage 3 setup checks, the next two terms cover the tensor reductions at receivers and their later internal group combinations, and the last two cover the terminal fold and the application batching before the root fold.

For a transcript-bound profile $(\lambda_{\mathsf{FS}}, Q_{\max})$, the schedule's Fiat–Shamir error budget is

$$\varepsilon_{\mathsf{FS}}(\mathsf{sch}; Q_{\max}) := (Q_{\max}+1)\varepsilon_{\mathsf{int}}(\mathsf{sch}) \leq 2^{-\lambda_{\mathsf{FS}}}. \tag{195}$$

Admissibility requires this inequality for the schedule's exact list of challenge rounds and the resulting error upper bound.

From One Fold to Fiat–Shamir Knowledge Soundness Theorem 10.40 iterates the one-fold extraction of Theorem 10.31 backward from the terminal on a complete transcript tree, with knowledge error $\varepsilon_{\mathsf{int}}(\mathsf{sch})$. The checkpoint structure of Proposition 10.44 and Theorem 10.45 then replaces interpolation at the strong checks by tests, which gives the Fiat–Shamir theorem (Theorem 10.46). We first fix the security model and the cost of the extractors.

Security model. The Fiat–Shamir reduction uses the classical random-oracle model and the classical Q-query checkpoint extractor of Theorem 10.17. The oracle includes the separately indexed programmable coordinate streams of Lemma 10.2; their queries count toward the same public query budget, and their lazy bit expansion is the ideal-XOF model of Section 10.1. The independent setup-expansion restriction $\mathcal{H}_{\mathsf{setup}}$ may be queried to recompute public parameters, but those queries do not enter the $Q+1$ forking loss. An admitted schedule carries a public pair $(\lambda_{\mathsf{FS}}, Q_{\max})$ in its transcript-bound digest. The concrete bit claim applies only to adversaries making at most $Q_{\max}$ queries to $\mathcal{H}_{\mathsf{FS}}$; Theorem 10.46 still gives an explicit bound in the actual query count $Q \le Q_{\max}$.

Sizing the Module-SIS inventory against quantum attacks does not make this a QROM reduction; a QROM Fiat–Shamir theorem for this multi-round, coordinate-wise extraction tree is outside the scope of this paper.

Extractor running time. Two tree sizes measure the extractors: the complete tree of Theorem 10.40 and the smaller checkpoint tree of the Fiat–Shamir extractor.

Complete tree. Let $\mathsf{Rounds}(\mathsf{sch})$ list the public-coin challenge stages of the actual protocol: every shared tensor-reduction prefix, public claim combination, fold, range stage, setup-product sum-check, and terminal. A vector sampled together is one stage. Stage r records its coordinate spaces $S_{r,u}$ and interpolation arities $k_{r,u}$ for $u \in U_r$, and whether it uses a flat coordinate-wise family or a nested interpolation tree. With the stage factor b_r of Lemma 10.14, define

$$\mathsf{Tree}(\mathsf{sch}) := \prod_{r \in \mathsf{Rounds}(\mathsf{sch})} b_r. \tag{196}$$

The stage factors are:
- $1 + \sum_g W_{h,g}$ for a fold, which is flat;
- Λ for a normalized affine combination of Λ claims, tensor or evaluation-trace;
- 2^{κ}, $2^{\mu'}$, and $2^{\lceil \log_2 n_{\mathsf{rel}} \rceil}$ for the nested stages of tensor row batching, the range point, and relation-row batching;
- $D+1$ for a scalar degree-D sum-check challenge, whose successive protocol rounds already form a nested tree. The binary-support check reuses the final range-stage tree and adds no nested tree.

Setup boundaries keep their separate internal claim-combination stages. The companion error of Lemma 10.14 is the sum of the coordinate charges already in (194).

Checkpoint tree. The Fiat–Shamir extractor branches only at the fold challenges and at the checkpoints of Proposition 10.44 and Theorem 10.45: the ring-reduction challenge of every nonterminal level and the first challenge of an application segment. For Akita's own batching, that challenge is ζ_{batch}, present when the root combines more than one claim. With $\chi_{\mathsf{root}} \in \{0,1\}$ indicating an application checkpoint, define

$$\mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch}) := 2^{|\mathsf{Lev}_{\mathsf{nt}}(\mathsf{sch})| + \chi_{\mathsf{root}}} (1 + B_{\mathsf{term}}) \prod_{h \in \mathsf{Lev}_{\mathsf{nt}}(\mathsf{sch})} \Big(1 + \sum_{g \in \mathcal{G}_h} W_{h,g}\Big). \tag{197}$$

The power of two counts one additional extraction per checkpoint, $1 + B_{\mathsf{term}}$ is the terminal fold factor, and the product is over the nonterminal fold factors. Both trees contain the fold factors, and the complete tree also has a range-point factor $2^{\mu'} \ge 2$ at every nonterminal level and a factor $\Lambda \ge 2$ at a batching root, so $\mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch}) \le \mathsf{Tree}(\mathsf{sch})$. Admissibility bounds $\mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch})$ by (142), a polynomial in $(\lambda, N_{\mathsf{exp,max}})$ for the total explicit instance-and-witness length $N_{\mathsf{exp,max}}$, not in its logarithmic variable count $\mu_{\max}$. Without that bound, the error expression remains valid but the extractor need not run in expected polynomial time.

Fiat–Shamir extractor. For a classical Fiat–Shamir adversary $\mathcal{A}$ making at most Q queries to $\mathcal{H}_{\mathsf{FS}}$, let $T_{\mathsf{FS}}(\mathcal{A}, Q, \mathsf{sch})$ be the expected running time of the extractor of Theorem 10.17 for the checkpoint structure used in the proof of Theorem 10.46: that of Proposition 10.44 when $\chi_{\mathsf{root}} = 0$ and of Theorem 10.45 when

$\chi_{\mathsf{root}} = 1$. With $\widehat{K} = \mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch})$, it runs the prover and verifier at most $\widehat{K}+(Q+1)(\widehat{K}-1) \le (Q+2)\,\mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch})$ times in expectation. For each rejection-sampled fold coordinate, it programs a conditional sampler tape using Lemma 10.2. Define

$$T_{\mathsf{filter}}(\mathsf{sch}) := \max_{r\in\mathsf{Rounds}(\mathsf{sch})} \sum_{u\in U_r} \frac{L_{r,u}}{\underline{a}_{r,u}}, \tag{198}$$

where an unfiltered challenge has $L_{r,u} = \underline{a}_{r,u} = 1$. Each run also does polynomial-time work: transcript checks, decoders and relation tests at the checkpoints, and deterministic algebraic extractors at the folds. Consequently,

$$\begin{aligned} T_{\mathsf{FS}}(\mathcal{A}, Q, \mathsf{sch}) \le{}& (Q+2)\cdot \mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch}) \\ &\cdot \mathsf{poly}\big(\lambda, |\mathsf{sch}|, N_{\mathsf{exp},\max}, T_{\mathcal{A}}, T_{\mathsf{ver}}(\mathsf{sch}), T_{\mathsf{filter}}(\mathsf{sch})\big), \end{aligned} \tag{199}$$

where the polynomial bounds the work per run and depends on the chosen classical rewinding implementation. With (142), this expected-time bound is polynomial in the admitted security and explicit-instance parameters.

Module-SIS algorithms. For each matrix instance I, the MSIS algorithm of Lemma 10.20(i) embeds its challenge into the I-th setup view, runs this extractor together with the collision computation, and keeps the output only if the collision names matrix I. Its expected running time is

$$T_{\mathcal{B}_I} \le T_{\mathsf{FS}}(\mathcal{A}, Q, \mathsf{sch}) + T_{\mathsf{embed}}(I) + T_{\mathsf{coll}}(I), \tag{200}$$

where $T_{\mathsf{coll}}(I)$ is the deterministic cost of subtraction, cross-multiplication of weak openings, or compression-chain descent for that matrix.

Theorem 10.40 (Interactive extraction for a fixed schedule). *Fix an extraction-admissible schedule* sch. *Given a complete accepting transcript tree whose challenges form the mixed family prescribed by* $\mathsf{Rounds}(\mathsf{sch})$, *a deterministic extractor returns either:*

(i) *an Akita batch witness satisfying* $\mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{batch}}$ *from* (173)*; or*

(ii) *a short nonzero kernel vector for one matrix view* $I \in \mathcal{I}(\mathsf{sch})$, *with norm at most* η_I.

If the same application commitment occurs in several opening groups, the extracted full-grid vector families for those occurrences agree; otherwise Corollary 10.25 gives alternative (ii). The extractor runs in time polynomial in $\mathsf{Tree}(\mathsf{sch})$ *and the explicit instance-and-witness length. Consequently, relative to the matrix-collision alternative, the interactive protocol specified by* sch *has knowledge error at most* $\varepsilon_{\mathsf{int}}(\mathsf{sch})$ *from* (194), *with an extractor whose expected running time is polynomial in* $\mathsf{Tree}(\mathsf{sch})$.

The theorem makes no assumption about the distribution of honest responses: a model-derived response bound enters only as a public bound that the verifier enforces in an accepting transcript.

Proof. **Terminal base case.** Apply Proposition 10.29 to the terminal subtree, which extracts the terminal source against the predecessor-bound inner state and binds its partials to the tensor-reduction output, and then Lemma 10.28 to the terminal's incoming recursive-witness group. This returns the ordinary base-field opening entering the terminal or a terminal $\mathbf{A}$-matrix collision, so the outgoing object of the last nonterminal fold is descendant-certified.

Backward induction. Suppose the ordinary opening produced by nonterminal fold h has been authenticated by its recursive suffix. For each folding-challenge child, run the strong extractors below that child from their authenticated leaves toward the folding node. If two openings under the same outgoing commitment give different source vectors, Corollary 10.25 returns a matrix collision. Otherwise the folding-challenge children form the descendant-certified family required by Theorem 10.31, which returns weak openings of the incoming source groups or a scheduled matrix collision.

At a coefficient-packing fold, the challenge-subring maps recover each incoming ordinary opening directly. At an evaluation-trace fold, Lemma 10.28 converts the extracted reduced claims into the incoming base-field openings at their native points. If setup offloading creates two input groups, the recursive-witness and setup commitments remain separate through this conversion; their fresh group-combination tree is extracted only afterward, and the setup-offloading relation fixes the setup group to the registered setup source. Repeating this step reaches the root.

Root claims. At the root, each extracted weak opening supplies, through its source projection, the algebraic decommitment of Definition 10.21. Applying $\mathsf{Dec}_{\mathsf{shape}_{\mathsf{com},g}}$ to $(\mathbf{s}_{g,c,i})_{i\in[B_g]}$ defines the decoded full commitment-grid vector. Lemma 10.33 separates the public batch and forces the multilinear extension of each decoded vector at $\mathbf{r}_g$ to equal $\phi_{g,c}v_{g,c}$. These are exactly the two conjuncts of (173); no padding-image claim is used.

Certified norms. Every accepted coefficient route supplies the certified response envelope used in its collision inventory. On a Euclidean route, Lemma 10.34 also forces the public squared norm to equal the sum of squares of the reconstructed integer response, and with the certified challenge-operator bound, Corollary 10.35 gives the scheduled Euclidean collision radius. Both arguments use only predicates proved in the accepting tree.

Error and running time. The tree extractor is deterministic. Applying Lemma 10.14 to the stage list gives the sum of the nonterminal, tensor-reduction, internal-group, terminal, and application-batch terms in (194), which proves the stated knowledge error and running time.

Remark 10.41 (What knowledge extraction does not claim). The source vector recovered by fold-challenge subtraction contains an inverse challenge difference and need not be the canonical, short gadget decomposition used by the honest commitment algorithm. The extractor returns a valid algebraic decommitment instead, and Corollary 10.25 makes its source projection computationally unique among successful extractions. For this reason Definition 3.7 uses a decommitment relation rather than re-running Commit and comparing its output state.

Recovering the canonical commitment state would require a separate canonicality or source-shortness argument; it follows neither from the bounded folded responses, since division by a unit need not preserve norm, nor from the strong–weak composition theorem. Akita does not claim it.

The rest of this subsubsection proves the Fiat–Shamir theorem with the checkpoint extractor of Theorem 10.17, which needs sibling transcripts only where it subtracts them. There are three steps: the opening suffix of a schedule has a checkpoint structure (Proposition 10.44); this structure composes with any application segment that ends with the row the suffix proves (Theorem 10.45); and Theorem 10.46 uses a one-challenge segment when the root batches several claims and no segment otherwise. A lemma on hashed transcript entries and two corollaries follow.

Checkpoint structure of a schedule. We split the protocol at the root fold challenge. The stages from this challenge on form the *opening suffix*; the stages before it reduce the application claims to one public row and are treated in Theorem 10.45. In the sense of Definition 10.16, the stages of the suffix have three types.

- *Extraction stages:* the fold challenge of every nonterminal level and of the terminal, with the flat coordinate-wise structure of Definition 10.30 and arity $1+\sum_g W_{h,g}$ at level h and $1+B_{\mathsf{term}}$ at the terminal. Every fold coordinate is read from its own coordinate stream, so the stage is indexed: coordinate u of group g has $k_{r,u}=2$ and $S_{r,u}=\mathcal{C}^{\mathsf{acc}}_{h,g}$, or $\mathcal{C}^{\mathsf{acc}}_{\mathsf{term}}$ at the terminal.
- *Checkpoint stages:* the ring-reduction challenge α of every nonterminal level. It immediately follows the fold challenge and is the first challenge after the successor payload of step 3 of Figure 4.
- *Plain stages:* the range, norm, row-batching, and sum-check challenges of steps 4 and 5, and the tensor-reduction and scalar-batching challenges of steps 0 and 2 at the receiving level.

The segment of the checkpoint at level h runs from its α to the last challenge before the fold challenge of level $h+1$ or of the terminal. Every plain stage of the suffix lies in the segment of one level.

For a nonterminal level h, a *successor source* σ is the data that an opening of the receiving level attaches to the payload bound in step 3, that is, the outgoing source that the recursive extractor authenticates. When the receiver is nonterminal, σ is the component of a source projection Src for that payload. When the receiver is the terminal, it consists of source blocks $(\mathbf{s}_i)_{i\in[B_{\mathsf{term}}]}$ with $\mathbf{A}_{\mathsf{term}}\mathbf{s}_i=\mathbf{t}_i$. The decoder fixed by the schedule maps σ to a table $\mathbf{w}_\sigma$ on $\{0,1\}^{\mu'}$ in the witness order of (76), on which the exact branch relation can be evaluated. On an offloaded edge, the receiver's setup group is read from the canonical setup source $\mathbf{S}^\flat$.

For a stage t in the segment of a nonterminal level, with work-predicate target g_t and challenge field E, put $\varepsilon_{\mathsf{st},t}:=2^{-g_t}N^{\mathsf{st}}_t/|E|$, where the numerator N^{st}_t is:

- $k_{\mathsf{ring}}-1$ for α;
- the number of coordinates for the range point τ_0, the row batch τ_1, and the tensor row batch;

– the degree of the round polynomial for a sum-check round: $d_s + 1$ in range-tree stage s, 3 in the fused relation sum-check, and 2 in Stage 3 and in the tensor reduction;
– $m_s - 1$ for the batch of range-tree frontier s, $N_{\mathsf{pair}} - 1$ for ζ_{pair}, and δ_f for η_{norm} on the digit-expanded route; and
– 1 for every other batching coefficient or vector of independent batching coefficients.

A stage that returns several of these challenges in one answer has the sum of their numbers. The next lemma supplies the bits and bad sets of item (vii) of Definition 10.16 on each checkpoint segment, with these errors.

Lemma 10.42 (Checks of a fixed successor source). *Fix an extraction-admissible schedule and a nonterminal level h. For every successor source σ and every stage input ξ from the input of the level's α through the input of the next fold challenge, there is a bit $D(\xi;\sigma)$, and for every stage t of the segment a set $\mathsf{Bad}_t(\xi;\sigma)$ of answers, such that:*

(i) *at the input of α, $D(\xi;\sigma) = 0$ exactly when no verifier check determined by ξ fails and $\mathbf{w}_\sigma$ satisfies the exact branch relation of ξ;*
(ii) *at the input of the next fold challenge, $D(\xi;\sigma) = 0$ exactly when no verifier check determined by ξ fails and the scalar row of Definition 10.22 for the receiving fold holds with every partial evaluation computed from σ and $\mathbf{S}^\flat$; at the terminal, the second condition is that every clear partial in ξ equals $\mathbf{a}_{\mathsf{term}}^\top \mathbf{G}_{b_{\mathsf{term}},M_{\mathsf{term}}} \mathbf{s}_i$;*
(iii) *if $D(\xi;\sigma) = 1$ at a stage-t input and the answer ρ is outside $\mathsf{Bad}_t(\xi;\sigma)$, then $D(\xi';\sigma) = 1$ at every next input ξ' that extends ξ by ρ; and*
(iv) *$\mathsf{Bad}_t(\xi;\sigma)$ has probability at most $\varepsilon_{\mathsf{st},t}$.*

The numbers N_t^{st} of the segment are termwise at most the numerators that (189), (190), $\varepsilon_{\mathsf{tensor},h+1}$, and (192) charge to the same challenges. The binary-support point term $\beta_h \mu'_h$ of (189) has no counterpart.

Proof. Pending claims. The *pending claims* at an input ξ are the claims that the protocol has stated or derived by ξ and has not yet checked. Every table in them is computed from σ, $\mathbf{S}^\flat$, and the public data in ξ. A batching coefficient not yet drawn is an indeterminate, so a batched claim is a polynomial identity in it. At the input of α the pending claims are conditions (i) and (ii) of the exact branch relation and, on a Euclidean route, the *norm obligation*

$$R_{\mathsf{dig}}(\sigma) \Longrightarrow \mathcal{E}_{\mathsf{int}}(\hat{\mathbf{z}}_\sigma) \le S_{\max},$$

where $R_{\mathsf{dig}}(\sigma)$ states that the response digits of $\mathbf{w}_\sigma$ lie in their certified ranges and $\hat{\mathbf{z}}_\sigma$ is the response that they encode. Together with condition (i), the norm obligation is equivalent to condition (iii). It refers to no challenge, so every stage before the norm claims leaves it unchanged. At the input of the next fold challenge the one pending claim is the scalar row, or the equalities of the clear partials at the terminal. In between they are the claims of Sections 6.1, 6.2, 7, 7.4 and 9.1 and of the receiver's tensor reduction.

The bit and the bad sets. Put $D(\xi;\sigma) = 1$ if a verifier check determined by ξ fails or a pending claim is false; this gives (i) and (ii). Let $\mathsf{Bad}_t(\xi;\sigma)$ be empty if $D(\xi;\sigma) = 0$ or if a check has failed, since a failed check stays failed on every extension. Otherwise fix an order on the pending claims and let $\mathsf{Bad}_t(\xi;\sigma)$ be the set described below for the first false one. Keeping $D = 1$ requires only that some pending claim remain false, so each bad set accounts for one false claim, not for every false claim.

It remains to show, for each kind of stage, that if a pending claim is false before the stage, then after it a pending claim is false or a check has failed, whenever the answer avoids a set of the stated size and whatever the prover sends next. A pending claim that the stage does not touch is unchanged.

Ring reduction. Suppose a ring row is false in its native ring. On the quotient-lift route, the lifted identities with the quotient entries of $\mathbf{w}_\sigma$ hold for at most $2d_{\max} - 1$ values of α by Lemma 7.1. On the quotient-free route, the reduced residual is a nonzero polynomial of degree below $d_{\max}$ by Theorem 7.4. In both cases at most $k_{\mathsf{ring}} - 1$ values of α are exceptional, by (186), and outside them a field row at α is false.

Multilinear batches. A family of identities indexed by a Boolean cube and batched with equality weights has a residual that is multilinear in the challenge vector. If one identity is false, the residual is nonzero and vanishes with probability at most the number of coordinates over $|E|$. This covers the range identity (80), which is false when an entry of $\mathbf{w}_\sigma$ lies outside the common alphabet, the row batch by τ_1, and the tensor row batch.

Sum-check rounds. If the claimed sum is false and the round check passes, the round polynomial differs from the true one, and the two agree on at most as many challenges as their degree. At any other challenge

the next round's claim is false. After the last round, the prover sends evaluations of tables derived from σ or $\mathbf{S}^\flat$. Either one of them is false and becomes a pending claim, or the verifier's final check fails.

Scalar batches. Claims combined by powers of one coefficient, or by independent coefficients, have a residual that is a polynomial in the coefficients. If a claim is false, the residual is nonzero, and its degree is at most the numerator N_t^{st}: one less than the number of claims in the frontier and digit-pair batches, the positive degree in η_{norm} noted after (93), and degree one for ζ_{norm}, γ, ζ_{bin}, and the normalized claim coefficients of the receiver.

The binariness claim. The binariness check on $\mathcal{I}_{\mathsf{bin}}$ has no challenge point of its own: it is evaluated at $\mathbf{r}_{\mathsf{virt}}$, whose coordinates are the challenges of the last range sum-check. Until that point is complete, the claim is pending as the residual $\sum_{x\in\mathcal{I}_{\mathsf{bin}}} \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{virt}}, x)\, \mathbf{w}_\sigma(x)(\mathbf{w}_\sigma(x)+1)$, a multilinear polynomial in the coordinates of $\mathbf{r}_{\mathsf{virt}}$ not yet drawn. It is nonzero exactly when an entry on $\mathcal{I}_{\mathsf{bin}}$ lies outside $\{-1, 0\}$, and a nonzero residual becomes zero for at most one value of the next coordinate. In a round of that sum-check, if the sum-check claim is false, the bad set is the one for sum-check rounds, and the false claim of the next round keeps $D = 1$ whatever happens to the residual. If the sum-check claim is true and the residual is nonzero, the bad set is this one value. Either way it has at most N_t^{st} elements, so the round is charged once, and the lemma has no term for the binary-support point.

Integer norms. The norm claims arrive before the last range sum-check. Suppose that the norm obligation is false, so that the response digits lie in their ranges and $\mathcal{E}_{\mathsf{int}}(\hat{\mathbf{z}}_\sigma) > S_{\max}$, and that no verifier check fails on the prefix extended by the norm claims. The norm obligation is replaced by the field identities that the claims state. On the direct route, both sides of (85) are integers in $[0, q)$, as in the proof of Lemma 6.3, and the decoded $\mathcal{E}_{\mathsf{resp}} \le S_{\max}$ differs from $\mathcal{E}_{\mathsf{int}}(\hat{\mathbf{z}}_\sigma)$, so the identity tested by the norm term is false. On the digit-expanded route, some claim $p_{t,a,a'}$ is false in $\mathbb{F}_q$: if all of them were true, Lemma 6.4 and Proposition 6.5 would make the verifier's reconstruction equal to $\mathcal{E}_{\mathsf{int}}(\hat{\mathbf{z}}_\sigma) > S_{\max}$, and the comparison with $\mathcal{E}_{\mathsf{resp}}$ would fail. This holds for every choice of the norm claims, so the transition has an empty bad set, although the claims may depend on all earlier challenges. The verifier's integer checks count as determined once the claims arrive, even when an implementation evaluates them at the end of Stage 1.

Level boundary. After the last sum-check of the level, the pending claims are the opening $\mathcal{W}_{h+1}$ and, on an offloaded edge, the opening $\mathcal{S}_h$ of (136); the latter is a claim about $\mathbf{S}^\flat$. An evaluation-trace receiver reduces them by the stages of Theorem 5.1, which are multilinear batches, sum-check rounds, and scalar batches as above, and the receiver's scalar batch then yields the scalar row. At the terminal, the verifier checks the trace equation on the clear partials against the tensor-reduction output, so a false output forces a clear partial to differ from that of σ.

Comparison with the error terms. The comparison is termwise: $k_{\mathsf{ring}} - 1$ against k_{ring}, one-shot batching bounds against coordinate-wise charges, and equal numbers elsewhere.

Remark 10.43 (Bad sets of pending claims). The proof maintains one invariant: a verifier check determined by the prefix has failed, or a pending claim about the fixed source is false. A stage that changes the first false claim has a bad set of the size below, and one that leaves it untouched has an empty bad set. Every table in a claim must be determined by the source, the prefix, and the auxiliary environment.

Transition of the first false claim	Bad-set size over $\lvert E\rvert$
equality-weighted batch over a Boolean cube	number of coordinates
batch with independent coefficients	1
batch by powers of one coefficient, n claims	$n - 1$
sum-check round of degree d	d
residual at a point drawn in later rounds	its degree in the next coordinate
challenge joint with a work predicate of g bits	multiply by 2^{-g}

The ring reduction and the integer norms are the two Akita-specific transitions, treated in the proof.

The opening suffix. This paragraph fixes the statement, the two relations, and the error used in Proposition 10.44. Let ξ_{f} be the input of the root fold challenge. It fixes the schedule, the root commitments $(\mathsf{cm}_g)_{g\in[G]}$ and points $\mathbf{r}_g$, the shared opening payload, the weights $\vartheta_{g,c}$, and the target v_{root}. For a *root weak opening* ω, a weak opening (Definition 10.22) of the root fold for the public data in ξ_{f}, put

$$\sigma := \mathsf{Src}(\omega), \qquad f^\star_{g,c} := \mathsf{Dec}_{\mathsf{shape}_{\mathsf{com},g}}\big((\mathbf{s}_{g,c,i})_{i\in[B_g]}\big), \tag{201}$$

and write $\mathsf{Dec}(\sigma)$ for the family of these vectors. The root uses coefficient packing (Equation (127)), with $e_{g,c,i} = \mathcal{L}_g(\mathbf{G}_{b_g,M_g}\mathbf{s}_{g,c,i})$; with the direct scalar opening (59), this gives $\mathcal{V}_{g,c}((e_{g,c,i})_i) = \widetilde{f^{\star}_{g,c}}(\mathbf{r}_g)$. The scalar row of a root weak opening is therefore

$$\mathsf{Row}_{\xi_\mathsf{f}}(\sigma): \qquad \sum_{g\in[G]}\sum_{c\in[n_{\mathsf{clm},g}]} \vartheta_{g,c}\,\widetilde{f^{\star}_{g,c}}(\mathbf{r}_g) = v_{\mathsf{root}}, \tag{202}$$

which depends on ω only through σ.

The *row-opening relation* has statement ξ_f and witness σ:

$$\mathsf{R}_{\mathsf{row}}(\xi_\mathsf{f};\sigma) = 1 \iff \bigwedge_{g\in[G]} \mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{com}}(\mathsf{pp},\mathsf{cm}_g,(f^{\star}_{g,c})_c,\sigma_g) = 1 \ \wedge\ \mathsf{Row}_{\xi_\mathsf{f}}(\sigma). \tag{203}$$

Every structure below has the binding relation whose witnesses are kernel vectors of the scheduled matrix views:

$$\begin{aligned}\mathsf{R}_{\mathsf{bind}}(\mathbf{S};I,\mathbf{x}) = 1 \iff\ & I\in\mathcal{I}(\mathsf{sch}),\quad 0\neq\mathbf{x}\in R_{q,d_I}^{m_I},\\ & \mathbf{M}_I(\mathbf{S})\mathbf{x} = 0,\quad \|\mathbf{x}\|_{p_I,\mathrm{coef}} \le \eta_I.\end{aligned} \tag{204}$$

Without application batching, write $\mathsf{Tree}^0_{\mathsf{ck}}(\mathsf{sch})$ for the value of (197) at $\chi_{\mathsf{root}} = 0$, and

$$\varepsilon^0_{\mathsf{int}}(\mathsf{sch}) := \varepsilon_{\mathsf{int}}(\mathsf{sch}) - \varepsilon_{\mathsf{batch}}(\mathsf{sch}) \tag{205}$$

for the interactive error.

Proposition 10.44 (Checkpoint structure of the opening suffix). *Fix an extraction-admissible schedule* sch *under the idealized setup. View the opening suffix as a protocol whose statement is the root fold input* ξ_f*, with any public weights and target in* ξ_f*. With the stage types above, it has a checkpoint structure for* $\mathsf{R}_{\mathsf{row}}$ *and* $\mathsf{R}_{\mathsf{bind}}$ *such that:*

(i) *a first-stage certificate is a prefix through* ξ_f *that passes every verifier check it determines, together with a root weak opening* ω *for it in which repeated occurrences of one commitment have equal source projections, and* Wit *returns* $\mathsf{Src}(\omega)$*;*

(ii) *the extraction stages are indexed, with stage errors* $\varepsilon_{\mathsf{cw},h}$ *at level* h *and* $\varepsilon_{\mathsf{term}}$ *at the terminal;*

(iii) *every other stage* t *has the error* $\varepsilon_{\mathsf{st},t}$ *of Lemma 10.42; and*

(iv) *the tree size is* $\mathsf{Tree}^0_{\mathsf{ck}}(\mathsf{sch})$*, and the stage errors sum to*

$$\varepsilon^0_{\mathsf{ck}}(\mathsf{sch}) := \varepsilon_{\mathsf{cw}}(\mathsf{sch}) + \sum_t \varepsilon_{\mathsf{st},t} \le \varepsilon^0_{\mathsf{int}}(\mathsf{sch}). \tag{206}$$

The structure reads the weights only as public data; it makes no assumption on how they were chosen.

Proof. We specify the data of Definition 10.16.

Certificates. Consider a stage after the checkpoint of level $h-1$, or the first stage when $h = 0$, and no later than the fold challenge of level h. A certificate at such a stage consists of the transcript through the message before that fold challenge, together with a weak opening of every incoming group of level h for this transcript, in which a setup group has the canonical source projection. For the terminal, the weak opening is replaced by source blocks with $\mathbf{A}_{\mathsf{term}}\mathbf{s}_i = \mathbf{t}_i$ that reproduce the clear partials, together with their unit challenge differences and bounded products.

In every certificate, the recorded transcript passes each verifier check that it determines; the maps below preserve this, because the transcript of their output is the common prefix of the transcripts of their inputs. A certificate at the end of the protocol is an accepting transcript. The certificates at a checkpoint are those of the next stage that pass its test, by item (v) of Definition 10.16.

Terminal fold. Apply the subtraction step in the first paragraph of the proof of Proposition 10.29 to $1 + B_{\mathsf{term}}$ accepting transcripts with a common prefix. It uses no other transcript and returns the terminal source blocks with their partials, or a short $\mathbf{A}$-matrix collision.

Nonterminal folds. The children are certificates of the checkpoint that follows the fold, so each passes the verifier checks on its prefix and carries a successor source whose table satisfies the exact branch relation of that child. Corollary 10.32 returns a weak opening of the incoming groups or a kernel vector. If the level

has a setup group, compare its source projection with the canonical reference occurrence; if one application commitment occurs in several root groups, compare their source projections. A disagreement gives a kernel vector by Corollary 10.25.

Both combiners use the common prefix and one sibling per coordinate, so each extraction stage is flat with $k_{r,u} = 2$. The subtraction cancels the other coordinates inside the fold-evaluation row of each group, which carries no weight, so it recovers every source block without inverting a weight.

Checkpoints. At the α of level h, the candidate of a certificate is its successor source, and the test is the exact branch relation of its decoded table for the certificate's prefix, including the integer response bound (iii); it reads no norm message. Two certificates with the same input at α contain the same successor payload. If their sources differ, Corollary 10.25 gives a kernel vector. When the receiver is the terminal, the two block families have the same image under $\mathbf{A}_{\mathsf{term}}$, and clearing their unit challenge differences as in the proof of Proposition 10.29 gives one.

Segment soundness. Lemma 10.42 supplies D and Bad for item (vii) of Definition 10.16. Condition (a) is item (i) of the lemma, and condition (b) is its items (iii) and (iv). For condition (c), no verifier check fails on the transcript of a certificate after α, and its weak opening satisfies the scalar row with the partial evaluations of its own source blocks and of the canonical setup source; this is item (ii).

Witness. A first-stage certificate is a root weak opening ω for a passing prefix through ξ_{f}. By Definition 10.22, ω satisfies the scalar row with $e_{g,c,i} = \mathcal{L}_g(\mathbf{G}_{b_g,M_g}\mathbf{s}_{g,c,i})$, which is $\mathsf{Row}_{\xi_{\mathsf{f}}}(\mathsf{Src}(\omega))$ by (202), and $\mathsf{Src}(\omega)$ is an algebraic decommitment of every root commitment to its decoded vectors. Hence $\mathsf{Src}(\omega)$ satisfies (203). The comparison at the root fold gives equal source projections for repeated commitments.

Size and errors. There are $|\mathsf{Lev}_{\mathsf{nt}}|$ checkpoints, and the extraction stages have the fold factors as arities, which gives $\mathsf{Tree}^0_{\mathsf{ck}}(\mathsf{sch})$. The errors of the extraction stages sum to $\varepsilon_{\mathsf{cw}}(\mathsf{sch})$ by (188). By the last part of Lemma 10.42, the remaining errors are at most the remaining terms of (194) other than $\varepsilon_{\mathsf{batch}}(\mathsf{sch})$.

Composition with an application segment. An application commits to vectors with Akita, runs a public-coin protocol about them, and reduces its claims to one row (202), which the opening suffix proves. Akita's own public batching is the case of a one-challenge segment. The composite protocol Π has three parts:

1. a first message with the statement x, the schedule, the root commitments $(\mathsf{cm}_g)_{g\in[G]}$, and possibly the root opening payload;
2. an *application segment*: the stages from a first challenge ℓ_0 to the last stage before the root fold challenge, with the prover messages between them (including the root opening payload if the first message omits it); it ends at ξ_{f}, which fixes the points, weights, and target of (202);
3. the opening suffix from ξ_{f}.

The application relation $\mathsf{R}_{\mathsf{app}}(x; f)$ relates the statement to a family f of decoded vectors for the root commitments and is decidable in time polynomial in the explicit instance-and-witness length. The composed relation is

$$\mathsf{R}(x;\sigma) = 1 \iff \bigwedge_{g\in[G]} \mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{com}}(\mathsf{pp}, \mathsf{cm}_g, (f^\star_{g,c})_c, \sigma_g) = 1 \ \wedge\ \mathsf{R}_{\mathsf{app}}(x; \mathsf{Dec}(\sigma)) = 1. \tag{207}$$

Let W_{ℓ_0} be the set of tuples $\sigma = (\sigma_g)_{g\in[G]}$ of source projections with the layouts fixed by the commitment shapes in x. The auxiliary environment is $\mathsf{aux} = (\mathbf{S}, z, \mathcal{O})$, where $\mathbf{S}$ is the flat setup vector and $(z, \mathcal{O})$ is drawn independently of $\mathbf{S}$ from an efficiently samplable distribution; $\mathsf{R}_{\mathsf{app}}$, the bits, and the bad sets below may read aux.

Theorem 10.45 (Composition with an application segment). *Fix an extraction-admissible schedule* sch *and the composite protocol* Π *above. Let* Good *be an event on* $(z, \mathcal{O})$. *Suppose that for every* aux *with* $(z, \mathcal{O}) \in \mathsf{Good}$ *and every* $\sigma \in W_{\ell_0}$ *there are bits* $D(\xi;\sigma)$, *for* ξ *an input of an application stage or* $\xi = \xi_{\mathsf{f}}$, *and sets* $\mathsf{Bad}_t(\xi;\sigma)$ *of probability at most* $\varepsilon_{\mathsf{app},t}$, *for* t *an application stage, such that:*

(a) *if* $\mathsf{R}_{\mathsf{app}}(x; \mathsf{Dec}(\sigma)) = 0$, *then* $D(\xi;\sigma) = 1$ *at every input* ξ *of stage* ℓ_0 *with statement* x;
(b) *if* $D(\xi;\sigma) = 1$ *at a stage-*t *input* ξ *and the answer* ρ *is outside* $\mathsf{Bad}_t(\xi;\sigma)$, *then* $D(\xi';\sigma) = 1$ *at every next input* ξ' *that extends* ξ *by* ρ;
(c) *if* ξ_{f} *passes every verifier check that it determines and* $\mathsf{Row}_{\xi_{\mathsf{f}}}(\sigma)$ *holds, then* $D(\xi_{\mathsf{f}};\sigma) = 0$.

Then Π *has a checkpoint structure for* R *and* $\mathsf{R}_{\mathsf{bind}}$ *on every such* aux. *It consists of the structure of Proposition 10.44 on the suffix, a checkpoint at* ℓ_0 *with candidate* $\mathsf{Src}(\omega)$ *and test* $\mathsf{R}_{\mathsf{app}}$, *and plain stages at the other*

application stages. The stage errors are $\varepsilon_{\mathsf{app},t}$ *in the segment and those of Proposition 10.44 in the suffix, and the tree size is* $2\,\mathsf{Tree}^0_{\mathsf{ck}}(\mathsf{sch})$.

Consequently, let P^* *be a classical random-oracle prover for the Fiat–Shamir transform of* Π *that makes* Q *queries to* $\mathcal{H}_{\mathsf{FS}}$*, with the setup expanded as in Section 4.3. There is an extractor that runs* P^* *at most* $2(Q+2)\,\mathsf{Tree}^0_{\mathsf{ck}}(\mathsf{sch})$ *times in expectation, and for each* $I \in \mathcal{I}(\mathsf{sch})$ *an algorithm* $\mathcal{B}_I$ *with expected running time (200), such that in the experiment of Definition 10.15*

$$\Pr\big[b = 1 \wedge w \notin \mathsf{R}(x)\big] \le \Delta_{\mathsf{setup}} + \Pr[(z,\mathcal{O}) \notin \mathsf{Good}] + \mathbb{E}\Big[\sum_{a \in \mathsf{Addr}(J)} \epsilon(a)\Big] + \sum_{I \in \mathcal{I}(\mathsf{sch})} \mathsf{Adv}_I(\mathcal{B}_I), \tag{208}$$

where $\epsilon(a)$ *is the weight (159) of this structure. The address sum satisfies*

$$\mathbb{E}\Big[\sum_{a \in \mathsf{Addr}(J)} \epsilon(a)\Big] \le \varepsilon^0_{\mathsf{ck}}(\mathsf{sch}) + \sum_t \varepsilon_{\mathsf{app},t} + Q\,\epsilon_{\max} \le (Q+1)\Big(\varepsilon^0_{\mathsf{int}}(\mathsf{sch}) + \sum_t \varepsilon_{\mathsf{app},t}\Big). \tag{209}$$

Proof. The structure. Take the data of Proposition 10.44 on the suffix, let Y_{f} be its set of first-stage certificates, and for $y \in Y_{\mathsf{f}}$ let ω_y be its root weak opening. Every application stage after ℓ_0 is plain, so by item (iii) of Definition 10.16 every certificate set from ℓ_0+1 up to the root fold challenge is Y_{f}. At ℓ_0 put

$$\mathsf{Cand}_{\ell_0}(y) := \mathsf{Wit}(y) := \mathsf{Src}(\omega_y), \qquad \mathsf{Good}_{\ell_0}(y) := \mathsf{R}_{\mathsf{app}}\big(x; \mathsf{Dec}(\mathsf{Src}(\omega_y))\big).$$

The one added checkpoint doubles the tree size.

Binding. Two certificates with the same input at ℓ_0 have the same first message, hence the same root commitments and schedule; their points, claimed values, and opening payloads may differ. If their source projections differ, Corollary 10.25 gives a witness of $\mathsf{R}_{\mathsf{bind}}$.

Segment. The segment of ℓ_0 is the application segment, and its end input is ξ_{f}. Items (vii)(a) and (vii)(b) of Definition 10.16 are conditions (a) and (b). For item (vii)(c), let $y \in Y_{\mathsf{f}}$: its prefix through ξ_{f} passes its determined checks, and $\mathsf{Row}_{\xi_{\mathsf{f}}}(\mathsf{Src}(\omega_y))$ holds by the witness step of Proposition 10.44, so condition (c) gives $D(\xi_{\mathsf{f}}; \mathsf{Src}(\omega_y)) = 0$.

Witness. For y that passes the test, $\mathsf{Src}(\omega_y)$ satisfies the first conjunct of (207) by Proposition 10.44 and the second by $\mathsf{Good}_{\ell_0}(y) = 1$.

Setup. The setup-expansion and Fiat–Shamir inputs occupy disjoint domain-separation namespaces. Replace the expanded setup by a uniform flat vector $\mathbf{S} \leftarrow \mathbb{Z}_q^{N_{\mathsf{setup}}}$, and answer every setup-domain query, including page queries, with output sampled from its real distribution conditioned on decoding to the corresponding coordinates of $\mathbf{S}$. Given $\mathbf{S}$, these answers have their real conditional law, so the joint law of the setup and the setup-domain answers changes by at most the statistical distance Δ_{setup} between the decoded and the uniform setup vector. The answers are sampled once, so the setup oracle is a function of aux, and the extractor never reprograms it.

Extraction. The extraction stages are fold challenges. For every rejection-sampled coordinate, Lemma 10.2 supplies an exact conditional random-oracle tape with overhead (198); response-dependent nonce probes remain part of Q. This is the sampling hypothesis of Theorem 10.17, and no other stage is programmed on a prefix of its answer. Apply that theorem with the event $(z,\mathcal{O}) \in \mathsf{Good}$. With probability at most the sum of the second and third terms of (208), the verifier accepts and the output is in neither $\mathsf{R}(x)$ nor $\mathsf{R}_{\mathsf{bind}}$. The bound on the runs is that of the theorem with $\widehat{K} = 2\,\mathsf{Tree}^0_{\mathsf{ck}}(\mathsf{sch})$, and (209) is its second inequality with (206).

Module-SIS. Let $\mathcal{B}$ receive $\mathbf{S}$, sample $(z,\mathcal{O})$, run P^* and the extractor, and output the kernel vector when the extractor returns one. It satisfies the hypothesis of Lemma 10.20 with the one-schedule family, and part (i) of that lemma gives the algorithms $\mathcal{B}_I$ and the last term of (208). This bound holds without conditioning on Good. Adding the four terms gives (208).

The interface between the two parts is condition (c): the segment must end with the row that the suffix proves, and D need decide nothing else at ξ_{f}. Since the bits and bad sets depend on σ and the transcript, and a weak opening has partials $\mathcal{L}_g(\mathbf{G}_{b_g,M_g}\mathbf{s}_{g,c,i})$, the theorem does not depend on the relative order of the root opening payload and the application challenges; it does not use the payload-before-weights order of Section 5.4. An application whose statement is already a row needs no segment: it applies Proposition 10.44 and Theorem 10.17 directly, at tree size $\mathsf{Tree}^0_{\mathsf{ck}}(\mathsf{sch})$.

Theorem 10.46 (Fiat–Shamir knowledge soundness of Akita). *Fix an admissible schedule* sch *with transcript-bound profile* $(\lambda_{\mathsf{FS}}, Q_{\max})$*, and instantiate the public setup as in Section 4.3, including the registry entries selected by its offloaded edges, so that* $\Delta_{\mathsf{setup}} \le N_{\mathsf{setup}}\, q\, 2^{-w}$ *by (36). Let* P^* *be a classical random-oracle prover making* $Q \le Q_{\max}$ *queries to* $\mathcal{H}_{\mathsf{FS}}$*. There is an extractor that runs* P^* *at most* $(Q+2)\,\mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch})$ *times in expectation, with expected time (199), and for each* $I \in \mathcal{I}(\mathsf{sch})$ *an ordinary Module-SIS algorithm* $\mathcal{B}_I$ *with expected running time (200), such that, in the batch analogue of the knowledge experiment of Definition 3.7 with statement* $\mathsf{x}_{\mathsf{batch}}$*, the probability that the verifier accepts and the extractor returns no witness for* $\mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{batch}}$ *is at most*

$$(Q+1)\,\varepsilon_{\mathsf{int}}(\mathsf{sch}) + \Delta_{\mathsf{setup}} + \sum_{I \in \mathcal{I}(\mathsf{sch})} \mathsf{Adv}_I(\mathcal{B}_I). \tag{210}$$

The first term may be replaced by the address sum

$$\mathbb{E}\Big[\sum_{a \in \mathsf{Addr}(J)} \epsilon(a)\Big] \le \varepsilon_{\mathsf{int}}(\mathsf{sch}) + Q\,\epsilon_{\max}(\mathsf{sch}), \tag{211}$$

in which a fold coordinate of group g *at level* h *has weight* $1/|\mathcal{C}^{\mathsf{acc}}_{h,g}|$*, a terminal fold coordinate* $1/|\mathcal{C}^{\mathsf{acc}}_{\mathsf{term}}|$*, the input of* ζ_{batch} *the weight* $(\Lambda-1)/|E|$*, the input of any other stage* t *its error* $\varepsilon_{\mathsf{st},t}$*, and* $\epsilon_{\max}(\mathsf{sch})$ *is the largest of these weights. Admissibility (195) gives* $(Q+1)\varepsilon_{\mathsf{int}}(\mathsf{sch}) \le 2^{-\lambda_{\mathsf{FS}}}$*. If ordinary Module-SIS is hard for every instance in* $\mathcal{I}(\mathsf{sch})$ *and* Δ_{setup} *is negligible, the Fiat–Shamir transform of the assembled protocol is adaptively knowledge sound for* $\mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{batch}}$ *in the classical random-oracle model.*

Proof. Batching root. Suppose first that the root combines $\Lambda > 1$ claims. Apply Theorem 10.45 with the following data:
– the application segment is the single challenge $\ell_0 = \zeta_{\mathsf{batch}}$;
– $\mathsf{R}_{\mathsf{app}}(\mathsf{x}_{\mathsf{batch}}; f)$ holds when $\widetilde{f_{g,c}}(\mathbf{r}_g) = \phi_{g,c} v_{g,c}$ for all (g,c), and Good is the sure event;
– the weights are (71), and the target is (72).
With this $\mathsf{R}_{\mathsf{app}}$, (207) is (173).

Bits and bad sets. Put $D(\xi_{\ell_0};\sigma) = 1$ exactly when $\mathsf{R}_{\mathsf{app}}(\mathsf{x}_{\mathsf{batch}}; \mathsf{Dec}(\sigma)) = 0$, and $D(\xi_{\mathsf{f}};\sigma) = 1$ exactly when a check determined by ξ_{f} fails or $\mathsf{Row}_{\xi_{\mathsf{f}}}(\sigma)$ is false. When $D(\xi_{\ell_0};\sigma) = 1$, let $\mathsf{Bad}(\xi_{\ell_0};\sigma)$ be the set of ζ_{batch} for which the row holds; otherwise let it be empty. Conditions (a) and (c) hold by definition. For condition (b), the values $\widetilde{f^{\star}_{g,c}}(\mathbf{r}_g)$ are fixed by σ, and the points and claimed values are fixed by ξ_{ℓ_0}, so Lemma 10.33 bounds the bad set by $(\Lambda-1)/|E|$. The row contains no prover message sent after ζ_{batch}, so outside the bad set it is false at every next input.

Size and error. The tree size $2\,\mathsf{Tree}^0_{\mathsf{ck}}(\mathsf{sch})$ is $\mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch})$ with $\chi_{\mathsf{root}} = 1$. The total stage error is at most $\varepsilon^0_{\mathsf{int}}(\mathsf{sch}) + (\Lambda-1)/|E| = \varepsilon_{\mathsf{int}}(\mathsf{sch})$ by (206), which gives (211) and, since $\epsilon_{\max} \le \varepsilon_{\mathsf{int}}$, the first term of (210).

One claim. If $\Lambda = 1$, the weight is one and the target is $\phi_{0,0} v_{0,0}$, so $\mathsf{R}_{\mathsf{row}}$ is $\mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{batch}}$. Apply Proposition 10.44 and the setup, extraction, and Module-SIS steps of the proof of Theorem 10.45 with no application segment; the tree size is $\mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch})$ with $\chi_{\mathsf{root}} = 0$.

Budget. Finally, $(Q+1)\varepsilon_{\mathsf{int}} \le (Q_{\max}+1)\varepsilon_{\mathsf{int}} \le 2^{-\lambda_{\mathsf{FS}}}$ by admissibility.

As in Theorem 10.40, these bounds do not assume the response model. With one commitment containing one decoded full-grid vector, one opening claim, and no natural-domain padding obligation, $\mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{batch}}$ specializes to $\mathsf{Rel}_{\mathsf{eval}}$ in (27). If a surrounding application separately certifies that an extracted common-grid vector is the canonical padding of a smaller evaluation vector, (51) transfers the extracted padded claim to that smaller natural-domain polynomial. An application that uses only the restrictions to its used slots can instead define its witness by those restrictions and prove the corresponding opening reduction, as in Lemma I.1, with no global zero-padding predicate. Theorem 10.46 asserts neither completeness nor binding; these are separate properties of the PCS definition.

Hashed transcript entries. Some messages enter the Fiat–Shamir transcript only through a digest. The next lemma transfers every knowledge-soundness bound of this section to such a protocol, with an additive collision term. Let Π be a Fiat–Shamir protocol and let Π_{H} be obtained from it by replacing, at positions fixed by the schedule, a message m by a digest $\mathcal{H}_o(m)$ of a type o. Each $\mathcal{H}_o$ is a random oracle with ℓ_o-bit

output, independent of $\mathcal{H}_{\mathsf{FS}}$, of $\mathcal{H}_{\mathsf{setup}}$, and of the other types. It belongs to the auxiliary environment, so reruns use the same $\mathcal{H}_o$. Digests may be nested: m may itself contain digests. The encoding is injective and records digest types: a string determines whether it is a stage key or a hash input, and where its digest positions of each type lie.

The verifier of Π_{H} recovers every hashed message from the proof, by reading or recomputing it, and checks its digest. It accepts exactly when these checks pass and the verifier of Π accepts the *expanded* proof, in which every digest is replaced by its message. The two protocols have the same statement and relation. The terminal of Section 8.2 is the case of one type dig with $\ell_{\mathsf{dig}} = 256$ and no nesting: it replaces the last block $\mathbf{t}_i$ and the last partial e_i by digests, and its verifier recomputes both from (131). The recomputed values satisfy (128) and (129) by construction, so this verifier has the stated form.

Lemma 10.47 (Hashed transcript entries). *Let P_{H} be a classical prover for Π_{H} that makes at most Q queries to $\mathcal{H}_{\mathsf{FS}}$ and Q_o queries to $\mathcal{H}_o$, counting the queries of the verifier on its output. Let V_o bound the total number of type-o digest positions in all these queries. There is a prover P for Π that makes at most Q queries to $\mathcal{H}_{\mathsf{FS}}$, runs P_{H} once with $O(Q + \sum_o Q_o)$ table operations per nesting level, and outputs the same statement, such that*

$$\Pr[\Pi_{\mathsf{H}} \text{ accepts } P_{\mathsf{H}} \text{ and } \Pi \text{ rejects } P] \le \varepsilon_{\mathsf{hash}} := \sum_o \frac{\binom{Q_o}{2} + Q_o V_o}{2^{\ell_o}}. \tag{212}$$

Consequently every knowledge-soundness bound for Π in this section, including (208) and (210), holds for Π_{H} with $\varepsilon_{\mathsf{hash}}$ added, where Q is the count above. A prover P_{H} that makes Q_{H} queries of its own has $Q \le Q_{\mathsf{H}} + Q_{\mathsf{ver}}$, where Q_{ver} is the number of challenge addresses of one transcript, so the budget condition of Theorem 10.46 reads $Q_{\mathsf{H}} + Q_{\mathsf{ver}} \le Q_{\max}$. The term $\varepsilon_{\mathsf{hash}}$ is not multiplied by $Q + 1$. For the terminal digests, $V_{\mathsf{dig}} \le 2Q$ and

$$\varepsilon_{\mathsf{dig}}(Q, Q_{\mathsf{dig}}) := \frac{\binom{Q_{\mathsf{dig}}}{2} + 2Q\,Q_{\mathsf{dig}}}{2^{256}}. \tag{213}$$

Proof. The prover P. P runs P_{H}, forwards its $\mathcal{H}_o$ queries, and records them in a table. The *expansion* of a string replaces each digest that has a recorded preimage by that preimage, recursively, and is defined when every digest is replaced. When P_{H} queries $\mathcal{H}_{\mathsf{FS}}$ at a challenge address whose key has a defined expansion, P forwards the address with its key expanded, and with the same coordinate at an indexed stage, to its own oracle and returns the answer. On every other $\mathcal{H}_{\mathsf{FS}}$ query, an address whose key has an unopened digest or a string that is not an address, P returns a fresh uniform answer and repeats it on a repeated query. When P_{H} halts, P runs the verifier of Π_{H} on its output under the same rules and outputs the expanded proof.

Bad events. For each type o consider two events, which range over the verifier's queries as well. In the first, two distinct $\mathcal{H}_o$ queries return the same value; this has probability at most $\binom{Q_o}{2}2^{-\ell_o}$. In the second, a fresh $\mathcal{H}_o$ query returns a string that already occupies a type-o digest position of an earlier query to any oracle; at most V_o such strings exist, so this has probability at most $Q_o V_o 2^{-\ell_o}$.

Simulation. Suppose no event occurs. Then a digest has at most one recorded preimage, and a digest that was unopened when it appeared in a query is never opened later. The expansion of every query is therefore fixed when the query is made, and it is injective on the forwarded queries. Forwarded queries receive independent uniform answers from the oracle of P, and all other queries receive independent uniform answers sampled by P, so the answers seen by P_{H} are distributed as a random oracle.

Acceptance. If the verifier of Π_{H} accepts, every digest in the proof hashes the message that the verifier recovers. The first $\mathcal{H}_o$ query that returned such a digest was fresh, so by the second event it preceded every query that contains the digest. Hence every digest was opened before the first query that contains it, and its unique preimage is the recovered message. Every challenge of the accepted proof thus equals the answer of $\mathcal{H}_{\mathsf{FS}}$ on the corresponding stage key of the expanded proof, and Π accepts it.

Consequence. Apply the extractor for Π to P. An execution in which Π_{H} accepts and extraction fails is either one in which Π accepts and extraction fails, or one counted in (212). Since $\mathcal{H}_o$ is part of the auxiliary environment, P uses the same $\mathcal{H}_o$ in every rerun, and the extracted witness is a witness for the common relation. For the terminal, a stage key has at most two digest positions and a digest input has none, so $V_{\mathsf{dig}} \le 2Q$.

The $\mathbf{t}$ digest precedes the predecessor's ring-reduction and sum-check challenges, and the e digest precedes the terminal response nonce and fold challenges. These are the positions of the full blocks in Π, so the expansion preserves the order of messages and challenges. The transcript absorbs different bytes for a digest than for a block, so a Π_{H} proof and a Π proof for the same witness have different challenges. The lemma relates the two provers and does not claim that proofs can be converted after the fact.

Corollary 10.48 (Schedule-specific Fiat–Shamir ceiling). *For a fixed schedule and query budget, define its effective Fiat–Shamir strength by*

$$\lambda^{\mathsf{eff}}_{\mathsf{FS}}(\mathsf{sch}; Q_{\max}) := -\log_2((Q_{\max}+1)\varepsilon_{\mathsf{int}}(\mathsf{sch}))\,. \tag{214}$$

An admissible schedule may claim only $\lambda_{\mathsf{FS}} \le \lambda^{\mathsf{eff}}_{\mathsf{FS}}$. If its interactive error contains a nonzero term $c/|E|$, then

$$\lambda^{\mathsf{eff}}_{\mathsf{FS}} \le \log_2|E| - \log_2 c - \log_2(Q_{\max}+1). \tag{215}$$

This ceiling depends on every term of the error bound, not only on the fold-challenge families. Its concrete effect on the supported field profiles is enforced by the transition closure of Section 12.2.

Proof. The first claim is (195) after taking negative base-two logarithms. Since $\varepsilon_{\mathsf{int}}(\mathsf{sch}) \ge c/|E|$, the same calculation gives (215).

Corollary 10.49 (Failure modes of a multi-instance opening). *Fix a schedule and a classical random-oracle prover making Q queries. In the uniform-setup experiment, the probability that the verifier accepts but the extractor returns neither a witness satisfying $\mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{batch}}$ nor a short nonzero kernel vector for some $I \in \mathcal{I}(\mathsf{sch})$ is at most the address sum (211), hence at most $(Q+1)\varepsilon_{\mathsf{int}}(\mathsf{sch})$. Replacing the uniform setup by the concrete seed expansion adds at most Δ_{setup}.*

Remark 10.50 (Concrete accounting). The terms of Theorem 10.46 grow differently with the adversary. Assume that every $I \in \mathcal{I}(\mathsf{sch})$ satisfies $\mathsf{Adv}_I(\mathcal{B}) \le T_{\mathcal{B}}\,2^{-\lambda_{\mathsf{MSIS}}}$ for every algorithm $\mathcal{B}$ of expected running time $T_{\mathcal{B}}$. Let T_{run} bound the expected cost of one run of the prover and verifier with the extractor work attached to it, and also $T_{\mathsf{embed}}(I) + T_{\mathsf{coll}}(I)$. By (210), (211), and (200), the failure probability of Theorem 10.46 is at most

$$\varepsilon_{\mathsf{int}}(\mathsf{sch}) + Q\,\epsilon_{\max}(\mathsf{sch}) + \Delta_{\mathsf{setup}} + (Q+3)\,T_{\mathsf{run}}\,2^{-\lambda_{\mathsf{bind}}}, \tag{216}$$

where $\lambda_{\mathsf{bind}} := \lambda_{\mathsf{MSIS}} - \log_2\big(|\mathcal{I}(\mathsf{sch})|\,\mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch})\big)$.

Term	Source	Growth
$\varepsilon_{\mathsf{int}}(\mathsf{sch})$	stage errors (194)	constant
$Q\,\epsilon_{\max}(\mathsf{sch})$	largest address weight (159)	linear in Q
Δ_{setup}	seed expansion (36)	constant
$(Q+3)\,T_{\mathsf{run}}\,2^{-\lambda_{\mathsf{bind}}}$	Lemma 10.20(i)	linear in $Q \cdot T_{\mathsf{run}}$

If the interactive error and the largest weight are at most $2^{-\lambda}$, the Fiat–Shamir terms are at most $(Q+1)2^{-\lambda}$: the schedule gives λ bits before the query count. The binding term gives λ_{bind} bits before the product of the query count and the cost of one run. If the assumption holds only for strict running times, Lemma 10.20(ii) at the optimal cutoff replaces the binding term by $2\sqrt{(Q+3)\,T_{\mathsf{run}}\,2^{-\lambda_{\mathsf{bind}}} + |\mathcal{I}(\mathsf{sch})|\,T_{\mathsf{run}}\,2^{-\lambda_{\mathsf{MSIS}}}}$.

10.4 Schedules Chosen from a Public Family

This subsection extends completeness and knowledge soundness from one fixed schedule to a schedule that the prover chooses from a public family after declaring its sizes (Corollary 10.51 and Theorem 10.52), for applications that learn the lengths of the committed evaluation vectors only when a proof is produced. Security does not depend on how the schedule was found; it requires that the verifier bind the choice before the first challenge and that every schedule it may accept satisfy the same admissibility and security conditions.

Write $\vec{N} = (N_0, \dots, N_{G_0-1})$ for the declared evaluation-vector lengths of the G_0 application-supplied groups at the root. A public planning policy $\mathcal{P}$ fixes the admitted declarations, the per-group size limits, and

a finite schedule family. For an admitted declaration $\vec{N}$, let $\mathcal{S}_{\mathcal{P}}(\vec{N})$ contain the schedules that the verifier may accept. The complete accepted family is

$$\mathcal{S}_{\mathcal{P}} := \bigcup_{\vec{N}} \mathcal{S}_{\mathcal{P}}(\vec{N}),$$

over admitted declarations $\vec{N}$. The transcript binds $(\vec{N}, \mathsf{sch})$, together with the common $(\lambda_{\mathsf{FS}}, Q_{\max})$ profile, before the first protocol challenge. We consider two ways to specify $\mathcal{S}_{\mathcal{P}}(\vec{N})$:

1. *Canonical.* A public deterministic planner with fixed tie breaking defines

$$\mathsf{sch} = \mathsf{Plan}_{\mathcal{P}}(\vec{N}), \tag{217}$$

   and the accepted set is this singleton. The verifier recomputes the prescribed output.
2. *Transmitted.* The prover sends a schedule, and the verifier accepts any schedule within the finite policy that the independent validator approves. The verifier checks admissibility.

In neither case is the planner's search part of proof verification.

Completeness lifts directly to the accepted family, and the lifted statement keeps the status of the response bounds used to select each schedule.

Corollary 10.51 (Size-adaptive completeness). *Suppose that every schedule in $\mathcal{S}_{\mathcal{P}}$ satisfies the hypotheses of Theorem 10.9, and set*

$$\varepsilon^{\mathcal{P}}_{\mathsf{comp}} := \max_{\mathsf{sch} \in \mathcal{S}_{\mathcal{P}}} \varepsilon_{\mathsf{comp}}(\mathsf{sch}). \tag{218}$$

Assume that the honest declaration and accepted schedule are chosen independently of the expanded setup. Then the honest prover is accepted with probability at least $1 - \varepsilon^{\mathcal{P}}_{\mathsf{comp}}$.

If every response bound used by every schedule in $\mathcal{S}_{\mathcal{P}}$ is certified, this is an unconditional completeness statement. If some response bound is model-derived, the statement is conditional on (152) for every such response stage and every reachable honest transcript prefix of its schedule.

Proof. Condition on the honest declaration and schedule. By the independence hypothesis, the expanded setup retains its original distribution. This fixes one admissible schedule, so Theorem 10.9 applies. Its failure probability is at most the maximum in (218). The same conditioning shows why a model-derived bound must hold uniformly over the reachable prefixes of every schedule in the accepted family.

For knowledge soundness, a prover that chooses among accepted schedules may meet different commitment matrices. Their Module-SIS instances form the inventory $\mathbb{I}_{\mathcal{P}} := \mathbb{I}_{\mathcal{S}_{\mathcal{P}}}$ of the family $\mathcal{S}_{\mathcal{P}}$, which is finite because the public policy has finite choice sets and bounded recursion depth. The reduction of Lemma 10.20 allows the schedule to be chosen after seeing the setup, so the fixed-schedule extractor applies without conditioning the setup on the prover's schedule choice.

Theorem 10.52 (Size-adaptive knowledge soundness). *Fix the public policy $\mathcal{P}$ and its finite accepted family $\mathcal{S}_{\mathcal{P}}$ independently of the setup. Suppose the verifier:*

(i) *accepts only admitted declarations and schedules in $\mathcal{S}_{\mathcal{P}}(\vec{N})$, each satisfying Definition 9.2;*
(ii) *in the canonical case, requires $\mathsf{sch} = \mathsf{Plan}_{\mathcal{P}}(\vec{N})$;*
(iii) *in the transmitted case, independently validates the supplied schedule;*
(iv) *binds $(\vec{N}, \mathsf{sch})$ before the first challenge; and*
(v) *accepts only schedules satisfying (195) and (142) for the common public profile.*

Then the Fiat–Shamir batched opening with declared sizes is knowledge sound in the sense of Theorem 10.46. In particular, the extractor returns a witness to $\mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{batch}}$ for the source commitments and decoded vectors on the full grid specified by the schedule. Any natural-domain padding-image predicate required by the application remains a separate obligation, as in (173); applications that use only restrictions to the application's used slots may instead apply Lemma I.1.

For a prover making $Q \le Q_{\max}$ queries to $\mathcal{H}_{\mathsf{FS}}$, extraction fails with probability at most

$$(Q+1) \max_{\mathsf{sch} \in \mathcal{S}_{\mathcal{P}}} \varepsilon_{\mathsf{int}}(\mathsf{sch}) + \Delta_{\mathsf{setup}} + \sum_{I \in \mathbb{I}_{\mathcal{P}}} \mathsf{Adv}_I(\mathcal{B}_I). \tag{219}$$

The first term is at most $2^{-\lambda_{\mathsf{FS}}}$ *because every accepted schedule satisfies* (195)*. It may be replaced by the address sum of Remark 10.19, which is at most* $\max_{\mathsf{sch}} \varepsilon_{\mathsf{int}}(\mathsf{sch}) + Q \max_{\mathsf{sch}} \epsilon_{\max}(\mathsf{sch})$*. Equivalently, a common family claim must obey*

$$\lambda_{\mathsf{FS}} \le \min_{\mathsf{sch} \in \mathcal{S}_{\mathcal{P}}} \lambda^{\mathsf{eff}}_{\mathsf{FS}}(\mathsf{sch}; Q_{\max}).$$

Proof. Setup and schedule choice. First replace the setup expansion by one uniform flat vector, at statistical cost Δ_{setup}. Run the prover without conditioning on its declaration or schedule choice. Since an accepted transcript binds both values before its first challenge, the classical Fiat–Shamir extractor treats them as part of the prover's pre-challenge message. Every stage input therefore encodes the bound schedule, and the proof of Theorem 10.46 gives a checkpoint structure for each schedule in $\mathcal{S}_{\mathcal{P}}$, with total stage error at most $\varepsilon_{\mathsf{int}}(\mathsf{sch})$.

Extraction. Apply Theorem 10.17 in the form of Remark 10.19: each node of the extractor reads the bound schedule from the stage inputs of its first run and uses that schedule's structure. The failure probability is at most $\max_{\mathsf{sch}} \varepsilon_{\mathsf{int}}(\mathsf{sch}) + Q \max_{\mathsf{sch}} \epsilon_{\max}(\mathsf{sch})$, which is at most the first term of (219). The expected number of prover runs is at most $(Q+2) \max_{\mathsf{sch}} \mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch})$, and the common bound (142) makes the expected running time polynomial.

Module-SIS. Any remaining extraction failure produces a collision for an instance in $\mathbb{I}_{\mathcal{P}}$. Apply part (i) of Lemma 10.20, with the family $\mathcal{S}_{\mathcal{P}}$, to the algorithm that runs the prover and the extractor, which may select its schedule after seeing the setup. This gives the final sum in (219), and Δ_{setup} is the cost of the initial setup replacement.

The Module-SIS term covers every schedule that the verifier may accept, however the planner orders them. If $|\mathbb{I}_{\mathcal{P}}| \le I_{\max}$ and every instance in the union has advantage at most $2^{-\lambda_{\mathsf{inst}}}$, this term is at most $I_{\max} 2^{-\lambda_{\mathsf{inst}}}$; a concrete claim may instead sum the bounds for the distinct instances (Appendix G).

11 Response Chunking for Distributed Proving

Akita already parallelizes the arithmetic of one fold across the cores and accelerators of a single machine. A different protocol is needed when the polynomial itself no longer fits in one machine's memory. Then several machines hold consecutive parts of the source witness, and each part, together with the large objects derived from it, should stay on the machine that stores it.

Most of the fold already fits this storage model. Commitment images, evaluation contributions, and ring-relation numerators are sums over the stored pieces, and sum-check rounds can be generated with exchanges only at ownership boundaries. This is the strategy of LaBRADOR [18]: distribute the large relation witness, then aggregate only its short images and proof messages. The folded response is the one object for which this strategy fails, and response chunking is the protocol change that handles it.

We first explain why the response cannot be aggregated cheaply. We then define the chunked witness and the one global relation that authenticates it, and show how the prover distributes the recursive commitment and sum-check. The final subsection accounts for the extra cost and proves completeness and extraction for the chunked relation.

11.1 The Distributed-Response Obstruction

For clarity, we treat one polynomial and one opening claim; the multi-group relation of Section 5.6 applies the same construction to each group in the group order. Let the source contain N ring elements, choose $M := 2^{r_{\mathsf{pos}}}$ positions per block, and set

$$B := \lceil N/M \rceil . \tag{220}$$

The source blocks are $\mathbf{f}_i = (F_{iM+x})_{x \in [M]} \in R_q^M$ for $i \in [B]$, with zeros after the final input entry. As in the ordinary fold, each block is decomposed into $\mathbf{s}_i$, with inner image $\mathbf{t}_i = \mathbf{A}\mathbf{s}_i$ and opening-method partial e_i.

Suppose machine P_j stores the blocks in a consecutive interval $\mathcal{I}_j \subseteq [B]$. The short commitment and opening images are sums over these intervals, but the folded response

$$\mathbf{z} = \sum_{i \in [B]} c_i^A \mathbf{s}_i, \tag{221}$$

where $c_i^A = \iota(c_i)$ under coefficient packing and $c_i^A = c_i$ under evaluation trace, does not split this way. Machine P_j can compute only the partial response

$$\mathbf{z}^{(j)} := \sum_{i \in \mathcal{I}_j} c_i^A \mathbf{s}_i, \tag{222}$$

which occupies the complete response space although P_j holds only part of the source. At a balanced first fold this space is itself about square-root sized, so recovering (221) requires an all-reduce of several full-width responses.

Decomposing first and adding later does not help: even if each $\hat{\mathbf{z}}^{(j)}$ is a balanced digit decomposition of $\mathbf{z}^{(j)}$, carries prevent $\sum_j \hat{\mathbf{z}}^{(j)}$ from being one of $\sum_j \mathbf{z}^{(j)}$. We therefore keep the local responses as separate segments of the next witness.

11.2 The Response-Chunked Witness

The protocol fixes a number N_{chunk} of *response chunks*, equal to 1 or a power of two at most 64. We first let source ownership follow whole blocks; Section 11.4 extends this to the intervals produced by recursion. Chunk $j \in [N_{\mathsf{chunk}}]$ corresponds to the canonical block interval

$$\mathcal{I}_j := \left[\left\lfloor \frac{jB}{N_{\mathsf{chunk}}} \right\rfloor, \left\lfloor \frac{(j+1)B}{N_{\mathsf{chunk}}} \right\rfloor \right). \tag{223}$$

These intervals are ordered, contiguous, and differ in length by at most one, and every finer admitted partition refines every coarser one. When $N_{\mathsf{chunk}} > B$, some intervals are empty; their chunks stay in the authenticated layout with a zero honest response.

The schedule fixes N_{chunk} before the fold challenges, and the transcript binds the resulting witness layout. The parameter counts ordered response segments, not machines: one machine may process several chunks, or several devices may cooperate on one chunk, without changing the proof.

For each interval, the prover decomposes the local response (222) and keeps the opening and inner-image digits of the blocks in that interval. Chunk j therefore contributes the contiguous witness unit

$$\mathbf{w}_j := [\hat{\mathbf{z}}^{(j)} \mid \hat{\mathbf{e}}^{(j)} \mid \hat{\mathbf{t}}^{(j)}]. \tag{224}$$

Concatenating the $\hat{\mathbf{e}}^{(j)}$ and $\hat{\mathbf{t}}^{(j)}$ recovers the ordinary $\hat{\mathbf{e}}$ and $\hat{\mathbf{t}}$ segments, but every $\hat{\mathbf{z}}^{(j)}$ is response-sized. Ignoring the shared compression and ring-check witnesses, and assuming every chunk uses the scheduled response-digit depth of the unchunked $\hat{\mathbf{z}}$, the body has length

$$|\mathbf{w}_{\mathsf{chunk}}| = N_{\mathsf{chunk}} |\hat{\mathbf{z}}| + |\hat{\mathbf{e}}| + \left|\hat{\mathbf{t}}\right|, \tag{225}$$

so chunking adds $(N_{\mathsf{chunk}} - 1)|\hat{\mathbf{z}}|$ coordinates before the shared tail.

All chunks use one fold-challenge vector and one grinding nonce, and the nonce is accepted only if every chunk's response satisfies the scheduled bound. The machines may test a nonce in parallel, but they cannot choose different nonces for different chunks, so every retained response is tied to one transcript challenge.

The algebra below does not depend on the opening method. Our admitted schedules use $N_{\mathsf{chunk}} > 1$ only in the leading coefficient-packing folds, where the source witness is largest, with the coefficient-ℓ_∞ response route. Once the recursive witness is small enough to consolidate, the schedule sets $N_{\mathsf{chunk}} = 1$, and later evaluation-trace folds may use the selective Euclidean route of Section 6.2.

11.3 One Relation over Chunked Witness Columns

Both response equations of the fold are linear in $\mathbf{z}$, so we substitute $\sum_j \mathbf{z}^{(j)}$ and split each equation into per-chunk contributions. Adding their short images recovers the ordinary fold identities without collecting the full response on one machine.

For the fold-evaluation equation, let $\mathcal{P}_{\mathsf{fold}}$ be the linear map that evaluates one recomposed response:

$$\mathcal{P}_{\mathsf{fold}}(\mathbf{z}) := \begin{cases} \mathcal{L}(\mathbf{G}_{b,M}\mathbf{z}) & \text{under coefficient packing,} \\ \mathbf{a}^\top \mathbf{G}_{b,M}\mathbf{z} & \text{under evaluation trace,} \end{cases} \tag{226}$$

valued in the coefficient-packing carrier ring in the first case and in the ambient ring in the second. The chunked fold-evaluation row is the single aggregate identity

$$\sum_{j\in[N_{\mathsf{chunk}}]}\left(\sum_{i\in\mathcal{I}_j} c_i e_i - \mathcal{P}_{\mathsf{fold}}(\mathbf{z}^{(j)})\right) = 0, \tag{227}$$

where c_i is read in the challenge subring under coefficient packing and in the ambient ring under evaluation trace, as in Figure 5.

The inner-consistency rows aggregate in the same way: for every $a \in [n_A]$,

$$\sum_{j\in[N_{\mathsf{chunk}}]}\left([\mathbf{A}\mathbf{z}^{(j)}]_a - \sum_{i\in\mathcal{I}_j} c_i^A [\mathbf{t}_i]_a\right) = 0. \tag{228}$$

These rows constrain only the sum of the retained responses, not the local fold identity of each chunk. The global opening claim needs only the sum; accordingly, extraction bounds differences of aggregate responses (Section 11.6).

The commitment equations use the opening and inner-image digits assigned to each chunk. Only the outer commitment $\mathbf{B}$ is sliced; the opening commitment uses one unsliced map $\mathbf{D}$. Let $\mathbf{x}_{B,s}^{(j)}$ be the contribution of chunk j to outer slice s, and let $\mathbf{D}^{(j)}$ be the columns of $\mathbf{D}$ acting on $\hat{\mathbf{e}}^{(j)}$. The two commitment relations are

$$\mathbf{u}_s = \sum_{j\in[N_{\mathsf{chunk}}]} \mathbf{B}\mathbf{x}_{B,s}^{(j)} \qquad (s \in [S_B]), \tag{229}$$

$$\mathbf{v} = \sum_{j\in[N_{\mathsf{chunk}}]} \mathbf{D}^{(j)}\hat{\mathbf{e}}^{(j)}. \tag{230}$$

One $\mathbf{F}$ compression chain binds the stack $\mathbf{u} = (\mathbf{u}_s)_{s\in[S_B]}$ and one $\mathbf{H}$ chain binds $\mathbf{v}$; their intermediate digit vectors and public payloads are shared by all chunks.

The scalar opening row also remains one field identity. If $\boldsymbol{\omega}_{\mathsf{open}}^{(j)}$ is the restriction of the direct or trace opening weight to $\hat{\mathbf{e}}^{(j)}$, then

$$\sum_{j\in[N_{\mathsf{chunk}}]} \left(\boldsymbol{\omega}_{\mathsf{open}}^{(j)}\right)^{\top} \hat{\mathbf{e}}^{(j)} = v_{\mathsf{open}}, \tag{231}$$

where $v_{\mathsf{open}} = v$ under coefficient packing and $v_{\mathsf{open}} = \bar{v}$ after the tensor reduction under evaluation trace.

Together these equations form one relation whose matrix is a horizontal concatenation

$$\mathbf{M} = [\mathbf{M}_0 \mid \mathbf{M}_1 \mid \cdots \mid \mathbf{M}_{N_{\mathsf{chunk}}-1} \mid \mathbf{M}_{\mathsf{shared}}], \tag{232}$$

where $\mathbf{M}_j$ holds the columns acting on $\mathbf{w}_j$ and $\mathbf{M}_{\mathsf{shared}}$ those acting on the shared compression and ring-check tails. Every chunk contributes to the original row families and their original public targets. Quotient lifting therefore introduces one quotient per ordinary ring row, not one per chunk, and quotient-free checking introduces none; the $\mathbf{F}/\mathbf{H}$ compression rows also occur once.

11.4 Local Images and the Recursive Commitment

We now show how each machine computes its contribution from the data it holds. On the $\mathbf{B}$ side, two partitions coexist: the S_B outer slices $I_{B,s}$ of (43), fixed when the commitment was formed, and the N_{chunk} chunk intervals $\mathcal{I}_j$. Their boundaries need not coincide. Chunk j contributes to slice s exactly the blocks in

$$\mathcal{J}_{j,s} := \mathcal{I}_j \cap I_{B,s}. \tag{233}$$

It places their $\hat{\mathbf{t}}$ digits at their canonical columns in $\mathbf{x}_{B,s}^{(j)}$ and writes zero in the remaining columns. Every block belongs to exactly one chunk, so these vectors add up to the fixed slice input,

$$\mathbf{x}_{B,s} = \sum_{j\in[N_{\mathsf{chunk}}]} \mathbf{x}_{B,s}^{(j)}, \tag{234}$$

and applying $\mathbf{B}$ gives (229) with the commitment's original columns and slices.

For $\mathbf{D}$, let $\mathrm{inc}_{D,j}$ embed chunk j's opening digits at their canonical positions in $\hat{\mathbf{e}}$, so that $\mathbf{D}^{(j)} = \mathbf{D}\,\mathrm{inc}_{D,j}$. Each machine computes the short image $\mathbf{v}^{(j)} := \mathbf{D}^{(j)}\hat{\mathbf{e}}^{(j)}$ and its contribution to the scalar opening row; the images add up to (230).

Ring reduction preserves this aggregation. With quotient lifting, the machines add their contributions to each ordinary row numerator, and the aggregator subtracts the one shared target and forms the one quotient for that row. With quotient-free checking, the local products are added directly under the transpose-convolution weights.

To continue the recursion, the local witness units and the shared tail are committed in one fixed order. With the quotient and compression segments of Section 5.5, the next witness is

$$\mathbf{w}' = \left(\Big\Vert_{j\in[N_{\mathsf{chunk}}]} [\hat{\mathbf{z}}^{(j)} \mid \hat{\mathbf{e}}^{(j)} \mid \hat{\mathbf{t}}^{(j)}] \right) \| \hat{\mathbf{Q}}_{\mathsf{main}} \Big\Vert \Big\Vert_{a=1}^{L_{\mathsf{comp}}} [\hat{\mathbf{c}}_a \mid \hat{\mathbf{Q}}_a], \tag{235}$$

with all quotient segments absent under quotient-free checking and the scheduled zero alignment of Section 5.5 understood. With several groups, each chunk holds their units in the fixed group order. No machine materializes all of $\mathbf{w}'$: each supplies the contribution of its chunk unit to the target commitment, an aggregation machine supplies the shared tail, and the outer slices of the target commitment determine where these contributions are added. The verifier still receives one commitment payload and one opening claim, as in the single-chunk case.

Ownership in the next fold. Each chunk unit stays on its machine and the shared tail goes to the last machine, with ownership fixed before the next fold challenges are sampled. The resulting intervals can be unequal and can cut through the next source blocks. We allow such a block to remain split: if $P_{j,i}$ selects the ring positions of block i held by machine j, that machine forms

$$\mathbf{z}^{(j)} = \sum_i c_i^A P_{j,i}\mathbf{s}_i. \tag{236}$$

The selectors keep all digit lanes of each owned ring position, so $\sum_j P_{j,i} = I$ and the local responses still sum to the ordinary response. The opening and inner-image digits keep the canonical intervals $\mathcal{I}_j$: source holders send their short undecomposed image contributions to the designated owner, which adds them before digit decomposition. The witness order and aggregate verifier equations are therefore unchanged. Each machine tests its own response under the common nonce, and grinding exchanges only success indicators. Commitments and quotients aggregate short linear images, and the range-check factors are pointwise functions of the locally held witness, so the sum-check below starts with its private factors already in place.

Aligning ownership boundaries to ring elements moves at most $O(Jd_{A,\mathrm{next}})$ base-field coefficients for J machines. Balancing the intervals is also cheap, because the response segments have a common length and cancel from the balancing cost: with auxiliary length a per source block (summed over groups) and a shared tail of length H, balancing moves $O(J(a+H+d_{A,\mathrm{next}}))$ coefficients. This is $\widetilde{O}(1)$ in the source length when J, a, H, and $d_{A,\mathrm{next}}$ are polylogarithmic. Alternatively, the prover may keep unequal intervals whenever their local response bounds fit the schedule (Proposition 11.1). These costs concern consecutive folds on the same machines; the eventual consolidation onto fewer machines has its own communication cost.

11.5 Distributed Sum-Check and Verifier Work

The range and relation checks need more than added images: a worker must hold all factors of a product before evaluating it. We distribute the sum-check terms accordingly and exchange factor values only where adjacent ownership intervals meet.

Consider an integrand

$$G(f_1(\mathbf{x}), \ldots, f_t(\mathbf{x}), \mathbf{x}),$$

where the f_a are prover-supplied multilinear polynomials with 2^ℓ Boolean evaluations each, the explicit argument $\mathbf{x}$ accounts for public factors, and D bounds the degree of a round polynomial. At the start of

each round, every remaining Boolean index has one owner, which holds the current evaluations of all t polynomials at that index, with previous challenges substituted. Each owner holds a consecutive interval of indices, and public factors are computed from the common transcript. This placement lets a worker form pointwise auxiliary values, such as the range-check product-tree entries, where their source values are stored. Any redistribution needed to establish the initial placement counts toward the prover's communication.

Rounds eliminate the least significant remaining index bit first. For each Boolean suffix $\mathbf{v}$, the pair $(0,\mathbf{v}),(1,\mathbf{v})$ is assigned to the owner of its even endpoint; if the odd endpoint has another owner, that owner sends its t current values to the pair owner. The pair owner then holds both endpoints of every multilinear factor and can interpolate them at X. Worker j forms its local round polynomial

$$g_j(X) := \sum_{\mathbf{v}\ \text{assigned to}\ j} G\big(f_1(X,\mathbf{v}),\dots,f_t(X,\mathbf{v}),X,\mathbf{v}\big),$$

with all preceding challenge coordinates already fixed. The coordinator sends $\sum_j g_j(X)$ to the verifier and broadcasts the next challenge r, and each pair owner stores $f_a(r,\mathbf{v})$ as the new entry at index $\mathbf{v}$. The ownership condition therefore holds again in the next round, even when the intervals have unequal lengths.

Every suffix contributes to exactly one local sum, so the round polynomials and final factor evaluations agree with those of a single prover on the chunked witness. This applies to every range-tree layer and to the relation sum-check; multiplying separately folded, zero-extended local evaluation vectors would not give the required products.

If at most J nonempty ownership intervals remain in a round, at most $J-1$ pairs cross interval boundaries, and exchanging their factor values costs at most $t(J-1)$ field elements. Workers also send at most $J(D+1)$ round coefficients, and the coordinator broadcasts the challenge. Over ℓ rounds, communication is therefore $O(J(t+D)\ell)$ field elements, in addition to the initial redistribution and the final factor openings. These messages accompany the short commitment images and scalar claims; no full response vector is aggregated across machines.

Verifier work. All response chunks use the same public matrices $\mathbf{A},\mathbf{B},\mathbf{D}$, so the verifier still performs one setup scan (126). The N_{chunk} response segments give separate weights $\Omega_A^{(j)}$ to the $\mathbf{A}$ columns, which the verifier adds before reading the setup:

$$\Omega_A^{\mathsf{chunk}}(a) := \sum_{j\in[N_{\mathsf{chunk}}]} \Omega_A^{(j)}(a). \tag{237}$$

It then evaluates each setup coefficient once. The opening and inner-image segments partition the ordinary $\mathbf{D}$ and $\mathbf{B}$ inputs, so their setup contributions are partitioned rather than replicated. The only extra verifier work is forming and adding the N_{chunk} response-weight fragments.

11.6 Cost, Completeness, and Security

Response chunking does not speed up an isolated Akita proof. The prover forms and decomposes N_{chunk} full-width responses, commits to the larger witness of (225), and runs the remaining recursion over that witness, which may add sum-check rounds or another fold. This cost is worthwhile only when it avoids response-sized communication or admits a source witness that does not fit on one machine.

The schedule authenticates both N_{chunk} and the number of leading folds that keep chunks. The limit $N_{\mathsf{chunk}} \le 64$ bounds the fragmentation an untrusted prover can impose on the verifier.

For completeness, one nonce must work for all chunks at once. As in the ordinary fold, we condition on the honest transcript prefix before the nonce search and take a union bound over chunks.

Proposition 11.1 (Completeness of a response-chunked fold). *Fix a reachable honest transcript prefix and condition on it. In the balanced-digit response model, let σ_∞ bound the absolute values of source coefficients, let C_j bound the number of source blocks contributing to any one response position of chunk j, and set $J_+ := \{j \in [N_{\mathsf{chunk}}] : C_j > 0\}$; chunks with empty source intervals have an identically zero honest response. For each $j \in J_+$, let*

$$V_j := C_j \kappa_2^2 \sigma_\infty^2 \tag{238}$$

be the uniform coordinate-variance bound of Lemma K.3. Whole-block ownership gives $C_j = |\mathcal{I}_j|$. For (236), an interval of n_j source ring positions gives $C_j = \lceil n_j/M \rceil$, so balanced ring intervals give $C_j \le \lceil B/N_{\mathsf{chunk}} \rceil$. If each response has n_z base-field coefficients and chunk j uses a symmetric threshold t_j contained in its canonical representable interval of (345), then the one-shot failure probability is at most

$$\delta_{\mathsf{chunk}} := \Pr\left[\exists j \in J_+ : \|\mathbf{z}^{(j)}\|_\infty > t_j \;\middle|\; \boldsymbol{c}_h \neq \perp\right] \le \sum_{j \in J_+} 2n_z \exp\left(-\frac{t_j^2}{2V_j}\right). \tag{239}$$

The one-hot path applies its chunk-local response bound to the same source-position subsets. The bound is conditional on successful challenge construction and, for a filtered family, requires the accepted distribution to satisfy the premises of the dense or one-hot tail used. Since accepting a nonce requires all chunks to succeed, the schedule chooses the digit depth and retry limit against the whole right-hand side. If $\underline{a}_h$ is the certified challenge-construction success bound and the verifier admits N_{try} fresh nonce trials, an honest prover exhausts them with probability at most

$$(1 - \underline{a}_h(1 - \delta_{\mathsf{chunk}}))^{N_{\mathsf{try}}}. \tag{240}$$

If $\underline{a}_h(1 - \delta_{\mathsf{chunk}}) > 0$, unrestricted rejection sampling takes at most $[\underline{a}_h(1 - \delta_{\mathsf{chunk}})]^{-1}$ trials in expectation.

Proof. At each response position of a nonempty chunk, the ring-aligned selector keeps at most C_j independently challenged source ring elements. Apply the coefficient tail bound (303) with this count and take a union bound over the n_z base-field coefficients. A second union bound over J_+ gives (239); it does not require the chunk responses to be independent. The bounded challenge-sampler lemma and distinct random-oracle nonces give (240) and the expectation bound.

The aggregate response equations authenticate only the effective response

$$\mathbf{z}_\Sigma := \sum_{j \in [N_{\mathsf{chunk}}]} \mathbf{z}^{(j)}. \tag{241}$$

When the extractor compares two transcript branches that differ in one folding challenge, several chunks may differ. The range check bounds the difference in chunk j coefficient-wise by its certified bound $\Delta^{\mathsf{cert}}_{\mathsf{rsp},j}$, so the difference in $\mathbf{z}_\Sigma$ is bounded by

$$\Delta^{\mathsf{cert}}_{\Sigma} := \sum_{j \in [N_{\mathsf{chunk}}]} \Delta^{\mathsf{cert}}_{\mathsf{rsp},j}, \tag{242}$$

which equals $N_{\mathsf{chunk}}\Delta^{\mathsf{cert}}_{\mathsf{rsp}}$ when all chunks use the same certified alphabet and digit depth. This aggregate bound is the response-difference parameter of the **A** binding argument.

Theorem 11.2 (Extraction of a response-chunked fold). *Fix a nonterminal response-chunked fold whose relation is (232), and suppose its outgoing claim is descendant-certified in the sense of Definition 10.30. From a nested coordinate-wise fold family, a deterministic extractor outputs either a weak opening of the underlying unchunked source relation or a current-level matrix collision. In the weak opening, the response-difference parameter of Definition 10.22 is*

$$\bar{\beta}_\infty = \Delta^{\mathsf{cert}}_{\Sigma}. \tag{243}$$

*Consequently, if $\bar{\kappa}_1$ bounds the extracted challenge difference, the coefficient-norm **A** collision of Lemma 10.23 has radius at most*

$$\eta_{A,\mathsf{chunk}} = 2\bar{\kappa}_1\Delta^{\mathsf{cert}}_{\Sigma}. \tag{244}$$

For an unfiltered challenge family whose members have ℓ_1 norm ω, one may take $\bar{\kappa}_1 \le 2\omega$, and hence

$$\eta_{A,\mathsf{chunk}} = 4\omega\Delta^{\mathsf{cert}}_{\Sigma}. \tag{245}$$

*All **B**, **D**, **F**, and **H** collision radii remain those of the ordinary fold.*

Proof. Apply the strong extractors below each folding-challenge child. They recover the exact aggregate identities (227) through (231), together with the certified range of every response chunk. As in Theorem 10.31, a disagreement in any shared compression-chain opening already gives the corresponding matrix collision, so assume all such digits are common.

Fix a folding coordinate q and choose the two children $(q,0),(q,1)$ guaranteed by its coordinate-wise special-sound structure. They agree outside coordinate q. Write $\bar{c}_q$ for challenge $(q,1)$ minus challenge $(q,0)$ at that coordinate, and write $\mathbf{z}_j^{(q,1)}, \mathbf{z}_j^{(q,0)}$ for the two extracted responses of chunk j. Applying the subtraction of Theorem 10.31 to the horizontally concatenated relation gives

$$\iota(\bar{c}_q)\mathbf{s}_q = \sum_{j\in[N_{\mathsf{chunk}}]} \left(\mathbf{z}_j^{(q,1)} - \mathbf{z}_j^{(q,0)}\right). \tag{246}$$

The certified range of chunk j bounds its summand by $\Delta^{\mathsf{cert}}_{\mathsf{rsp},j}$ coefficient-wise, and the triangle inequality gives (243). The factor N_{chunk} enters here because the aggregate rows do not identify which chunk changed between the two branches.

The same subtraction in the aggregate $\mathbf{A}$ and fold-evaluation rows, followed by cancellation of the unit $\bar{c}_q$, recovers the ordinary inner commitment and partial-evaluation relations for $\mathbf{s}_q$. Repeating the argument for every folding coordinate produces a weak opening of the underlying source vector. Applying Lemma 10.23 with $\bar{\beta}_\infty = \Delta^{\mathsf{cert}}_{\Sigma}$ gives (244), and the unfiltered-family bound gives (245). Chunking neither duplicates the other semantic row families nor changes their certified alphabets, so their collision radii are unchanged.

The schedule-level guarantees then follow by substituting the chunked completeness bound, collision radius, and sum-check parameters.

Corollary 11.3 (Scheduled Akita security with response chunks). *Let* sch *satisfy the premises of Theorems 10.9 and 10.46, except that some leading coefficient-packing folds may use the response-chunked relation of this section. Make the following substitutions at every such level h:*

1. *let $\mathcal{R}_h$ in Definition 10.5 index the individual response chunks; Proposition 11.1 then proves the corresponding uniform conditional nonce-failure bound with $\delta_h = 1 - \underline{a}_h(1-\delta_{\mathsf{chunk}})$;*
2. *replace the ordinary response-difference envelope in the $\mathbf{A}$-role Module-SIS instance by $\Delta^{\mathsf{cert}}_{\Sigma,h}$, and on the admitted coefficient route set its collision radius according to (244), or equivalently (245) for an unfiltered challenge family;*
3. *compute $\varepsilon_h^{\mathsf{sch}}$ using the actual chunked witness length, sum-check round count, and ring-check degrees at level h.*

Then the completeness bound of Theorem 10.9 and the Fiat–Shamir knowledge-soundness bound of Theorem 10.46 hold without any additional error term. In particular, the advantage bound remains (210), with the modified $\mathbf{A}$ instance in $\mathcal{I}(\mathsf{sch})$ and the recomputed $\varepsilon_h^{\mathsf{sch}}$.

Proof. The first substitution is exactly the completeness accounting of Proposition 11.1. At the extraction stages of Proposition 10.44, used in the proof of Theorem 10.46, replace Theorem 10.31 at a chunked level by Theorem 11.2. The other extraction stages use the same unchunked weak-opening interface. The larger witness changes the scheduled sum-check parameters and may change successor layouts, but does not create another category of algebraic failure. At the chunked fold, the shared setup induces the same matrix views, with only the $\mathbf{A}$ collision radius enlarged according to (244). This gives the stated substitutions in (210).

This security statement treats all machines and the aggregator jointly as one prover for the chunked relation. With the aggregate coefficient envelope accounted for above, distribution requires no new hardness assumption. Response chunking provides no privacy, robustness, fairness, or security against malicious workers within that prover; these properties require a separate multiparty protocol.

12 Offline Planner and Parameter Selection

The preceding sections define a family of admissible Akita schedules. This section chooses one for a given input shape and resource budget. The difficulty is that a fold changes the problem faced by the next fold: its response, commitment digits, and any deferred setup opening all become part of the remaining work. We

use an offline planner to compare these consequences across the complete recursion. A separate validator checks the chosen schedule against the protocol and security conditions, so correctness does not depend on the search finding a good choice.

12.1 Parameter-Selection Approaches

The Greyhound reference implementation chooses parameters during proving and keeps a reduction when its proof and remaining witness are smaller than a direct reveal [68]. LaZer's newer LaBRADOR implementation instead generates a sequence offline through successive local size improvements [70]; the inspected RoKoKo artifact supplies concrete chains, with automatic selection listed as future work [88]. We compare complete schedules, including later setup-opening obligations, under objectives that prioritize verifier efficiency. We also validate the chosen schedule independently of the search. These distinctions concern the cited artifacts' selection strategies; they neither indicate limitations of the underlying proof systems nor guarantee that Akita improves every cost.

12.2 Choosing a Complete Schedule

A planning instance fixes the public protocol environment Env and the incoming opening shape $\mathcal{O}_0$. The environment specifies the field, available setup, supported protocol variants, certified challenge families, and concrete security targets. The root shape includes the geometry of every commitment already formed. Such a commitment cannot acquire a new block split, ring, rank, or compression map when it is opened. When provisioning a commitment together with its future openings, we may first compare root shapes and then fix the chosen one.

A finite planning policy $\mathcal{P}$ specifies which parameter choices to consider, a maximum recursion depth, resource limits, and an ordering of complete schedules. It can restrict choices for tractability without making those restrictions part of the protocol. A schedule records the selected folds and terminal, as in (17); admissibility is defined in Definition 9.2. Our guiding priority is verifier efficiency, followed by proof size and prover work. The relative importance of online work and stored setup can depend on the application, so the protocol does not prescribe numerical weights for these costs.

Deriving the continuation. We derive each candidate's complete successor using (76). A smaller response alone need not give a cheaper continuation: compression adds witness digits, and offloading adds a setup opening that the successor must check. The compression transition and opening obligations obey Definition 9.2; their widths are fixed by Sections 4.5 and 9.2. For each candidate ring dimension, we derive the padded matrix width and sufficient rank before comparing the costs in (247). Shared setup storage is the largest required prefix, not the sum of the prefix lengths. These dependencies are included before a candidate is scored.

Ring dimension and matrix cost. Consider a matrix instance I with L_I input field coefficients. At ring dimension d_I, these coefficients occupy m_I input ring elements, and a matrix of module rank n_I produces $T_I := n_I d_I$ output field coefficients. To isolate the effect of the dimension, temporarily hold L_I and T_I fixed and assume that no input padding is needed. The input width, number of setup coefficients, and number of ring products in a matrix-vector multiplication are then

$$m_I = \frac{L_I}{d_I}, \qquad n_I m_I d_I = \frac{T_I L_I}{d_I}, \qquad n_I m_I = \frac{T_I L_I}{d_I^2}. \tag{247}$$

Thus larger rings reduce both the stored matrix size and the number of ring products in this comparison. Each ring product also becomes more expensive, so these counts alone do not determine the running time. At module rank one, $n_I = 1$, the matrix contains exactly as many field coefficients as its padded input. This equality also helps account for the setup prefix carried by an offloaded fold (Section 9.2).

The required output size T_I can change with d_I: Module-SIS security also depends on the modulus, input width, and collision norm. Moreover, the rank is an integer, so output lengths come in multiples of d_I. Rounding a fixed coefficient target up to such a multiple adds at most $d_I - 1$ coefficients, but the security requirement itself must be recomputed at each candidate dimension. We therefore compare candidates using the exact padded input widths and secure ranks.

Challenge and inner-commitment dimensions. At a packing fold, a candidate first selects d_{car} and an audited challenge family at that dimension, then an A-ring dimension with $kd_{\mathsf{car}} \mid d_A$ (Section 4.4). Once d_{car} is fixed, increasing d_A enlarges neither the partial evaluations, which have kd_{car} base-field coordinates each, nor the packing quotient. We therefore choose the challenge family from d_{car} and compare compatible pairs (d_A, d_{car}). A larger A ring can reduce the required $\mathbf{A}$ rank without changing the challenge family. A smaller challenge subring can shorten the partial evaluations, but a heavier challenge can enlarge the response and the SIS collision bound. These effects must be evaluated together. The schedule fixes which response-certification routes are admissible (Section 9.3).

Comparing costs. We record proof length, verifier work, prover work, setup storage, and distributed communication as

$$\mathsf{cost}(\mathsf{sch}) = (C_{\mathsf{pf}}, C_{\mathsf{V}}, C_{\mathsf{P}}, C_{\mathsf{setup}}, C_{\mathsf{comm}}). \tag{248}$$

Each quantity is measured in fixed units for the planning instance. Per-fold messages and work add, while shared setup is counted once. The proof-length component uses the round-message counts in Table 4; endpoint values and fixed payloads are added separately. Let $\mathsf{Score}_{\mathcal{P}}$ denote the policy's ordering of these complete-schedule costs, including a rule for ties. Our selection problem is

$$\underset{\substack{\mathsf{sch}\text{ in the family specified by }\mathcal{P}\\ \mathsf{sch}\text{ admissible and within resource limits}}}{\text{minimize}} \quad \mathsf{Score}_{\mathcal{P}}(\mathsf{sch}). \tag{249}$$

Changing the cost model changes which admissible schedule we prefer. It does not change the protocol conditions that the schedule must satisfy.

12.3 Predicting the Next Fold

To compare continuations before producing a proof, we need more than their matrix dimensions. The next fold uses digits of the current response, and the number and size of those digits depend on its scheduled bounds. A bound that is unnecessarily large can therefore increase the cost of several later folds. A bound that is too small can make an honest prover exhaust its retries. The planner must account for this completeness cost alongside the transmitted bytes and arithmetic work.

At the root, Lemma K.3 and Theorem K.5 give rigorous response bounds under their stated challenge-distribution hypotheses. For later folds, we can propagate certified envelopes, or use a response model to propose tighter bounds. Our model keeps separate estimates for the response digits, opening digits, matrix images, and compression digits in the recursive witness. It tracks total squared norms together with full-ring averaged and local coefficient-moment estimates used to predict response peaks: a small total squared norm alone does not rule out a coefficient too large for the chosen digit depth. Appendix K.3 gives the model and the calculation used to propose joint coefficient and squared-norm bounds.

The model proposes bounds; it does not prove that the actual integer responses follow its Gaussian surrogate. A schedule using model-derived bounds therefore retains the history-conditional premise of (152): at every reachable honest prefix, all responses must pass jointly with the stated probability after successful challenge construction. The response analysis combines this premise with the certified challenge-sampler probability and retry budget, then derives the schedule-level conditional completeness bound (Proposition K.11 and Corollary 10.10). Separate per-response estimates require an additional joint argument, such as a union bound, and numerical calibration alone does not prove the premise.

The verifier always enforces the selected bounds, which are used for soundness. Certified alternatives, including Corollary K.9 and the digit squared-norm bound of (344), can replace modeled entries when their hypotheses hold. If they enlarge a bound, we propagate the change through the remaining schedule and recheck its security and cost.

12.4 Searching Across the Recursion

We use dynamic programming over the finite policy, comparing an admitted terminal with further folds at each state. A state retains the opening shape, depth, restrictions imposed by earlier folds, and pending obligations; each continuation retains its costs and feasibility data. Because setup is shared, we discard a

Table 4: Sum-check round-message accounting for one Akita fold. The degree column records the maximum degree of the summand in one sum-check variable. A standard degree-D message omits its linear coefficient and sends D field elements; an equality-factored range message omits its constant coefficient after removing the common equality factor.

Stage	Rounds	Degree	Format	Elements per round
Nonidentity shared tensor reduction	$m_{\max}$	2	standard	2
Single-claim nonidentity tensor reduction	$\ell - \kappa$	2	standard	2
Identity tensor reduction, including $k = 1$	0	–	none	0
Range product stage i, without norm fusion	μ'	$d_i + 1$	equality-factored	d_i
Final range stage with either norm route	μ'	$d_i + 1$	standard	$d_i + 1$
Fused relation, range binding, binary support, and optional norm-claim binding	μ'	3	standard	3
Setup-product sum-check	ν_S	2	standard	2

For the norm-fused row, $d_i \in \{2, 4\}$, while the direct and digit-expanded norm terms have degrees two and three. Fusion therefore changes the message format and adds one field element per existing round, but adds no round. A prescribed sum-check over zero variables is checked directly and has no round message; this includes a setup-product check with $\nu_S = 0$. An identity tensor reduction is omitted regardless of tail arity. At the terminal base proof, only its optional tensor-reduction prefix contributes sum-check round messages; no range or relation sum-check is run. The counts exclude endpoint values and other fixed messages, including tensor column partials, range-tree frontier claims, norm claims, final evaluations, nonces, and the outgoing recursive opening.

continuation only if it is infeasible or a retained replacement is feasible and no worse after every admissible preceding path. With exhaustive search, backward induction then gives the optimum in (249). Restricting candidates or retained continuations narrows or removes that guarantee, but does not change admissibility. The artifact specifies the concrete candidate sets and search limits [65,66].

12.5 Validating and Using Schedules

The validator rederives every successor from the public environment and frozen root shape, then checks Definition 9.2 and the resource limits. It uses the protocol definitions rather than the optimizer's cached shapes or security estimates. The Module-SIS estimator of Appendix G, the schedule-wide bounds of Theorem 10.46, and any conditional completeness premises retain their stated hypotheses. The validator also derives the largest admitted shared setup prefix, recomputes its setup-expansion error Δ_{setup}, and checks it against the common security profile.

Validation precedes proof verification: an opening binds the validated schedule and input declaration before its first challenge. Applications may retrieve a prevalidated schedule for their input sizes or supply a member of a fixed public family for validation. Existing commitments keep their shapes, and security must cover the whole selectable family as in Section 10.4. Neither validation nor proof verification requires repeating the search or establishing optimality.

12.6 Sum-Check Message Accounting

Table 4 lists the round messages of every sum-check in one fold, which the proof-length component of (248) counts.

13 Implementation and Evaluation

We evaluate Akita first as a standalone PCS and then as a commitment backend for Jolt. The standalone experiments separate the cost of creating a commitment from the cost of opening it, and compare verification time, proof size, and memory use. The Jolt experiment tests whether these tradeoffs improve a complete prover, where commitments are batched and the workload contains sparse as well as dense polynomials.

13.1 Implementation and Experimental Methodology

We implement Akita in Rust and integrate it into Jolt [66,11]. The implementation supports 32-, 64-, and 128-bit prime fields, with scalar, NEON, AVX2, and AVX-512 arithmetic backends. We use the schedules selected by the offline planner and accepted by its independent validator, without manual parameter adjustments. Planning and validation are excluded from the online measurements. The Akita Book describes the arithmetic and prover optimizations used by the implementation [65]. The implementation and benchmark artifacts are publicly available [66,69]. Speedup ratios use unrounded timing medians.

Workloads and metrics. Each standalone experiment measures a native commitment-and-opening workload; Appendix M specifies the single-polynomial and batched cases. We report commitment and opening times separately because a commitment can be reused for multiple openings. Verification time measures checking one opening; proof size measures the opening proof, excluding the commitment. We also report the communication of commitment and proof together, and measure memory as the peak resident set size of the complete commitment-and-opening process. Reusable setup is reported separately where the implementation exposes it; Greyhound includes key setup in its commitment time.

To compare polynomials over different fields, we measure input size by the number of bits committed,

$$B = N \log_2 |\mathbb{F}|,$$

where N is the number of coefficients and $\mathbb{F}$ is the coefficient field. We target $B = 2^{27}, 2^{29}, 2^{31}, 2^{33}, 2^{35}$ bits, choosing the closest supported power-of-two coefficient count. For a 32-bit field, this corresponds to $N = 2^{22}, 2^{24}, 2^{26}, 2^{28}, 2^{30}$. This is a nominal field-capacity measure, not the entropy of the sampled input. The schemes keep their native polynomial interfaces, so equal input sizes do not imply identical statements: they can have different numbers of variables and different opening fields. Binius64 and Flock measure native packed-field openings, excluding the tensor-algebra reduction from an original binary-subfield claim. Appendix M details these workload differences.

PCS benchmark software. We imported the hash-based and lattice-based baselines into a common benchmarking framework [69].

Hardware and threads. Standalone measurements use an AMD Ryzen 9 9950X with AVX-512, 186 GiB RAM, and a 168 GiB per-process memory limit. All verification measurements in this section use one thread. We also use one thread for lattice-PCS proving because the public Greyhound and RoKoKo implementations do not expose comparable end-to-end multithreaded provers. The hash comparison includes both one- and eight-thread proving. Jolt uses an Apple M4 Max, as described below. Sizes are reported in decimal KB (1000 bytes), except where a plot explicitly uses KiB (1024 bytes).

Scope of the comparisons. Security assumptions and parameter-selection conventions differ across the baselines. Greyhound uses a 128-bit quantum ADPS16 Euclidean-SIS policy, but its truncated-output Ring-SIS assumption is distinct from Akita's standard Module-SIS assumption. RoKoKo estimates its structured vSIS assumption with GSA/MATZOV-classical and states a 100-bit statistical soundness target [62, Section 9.3]. Its approximate low-norm challenge sampling has an estimated noninvertibility probability of about 2^{-94} [62, Section 9.1]. Appendix J.2 separates these current qualifications from the repaired implementation error. The hash implementations likewise differ in their security targets and proximity analyses. We therefore compare the measured implementations without treating every runtime ratio as an equal-security result. Appendix J discusses the security distinctions.

13.2 Comparison with Lattice-Based PCSs

We compare Akita with Greyhound and RoKoKo. Akita and Greyhound use the same 32-bit field. RoKoKo's supported native instances use a roughly 50-bit field, giving 25/16 times the nominal bit capacity at the same coefficient count; we report their measured costs without rescaling them. Its quadratic-extension opening field has approximately 100 bits, whereas Akita opens over a degree-4 extension of its 32-bit coefficient field. For Akita, we distinguish direct setup evaluation from setup offloading, which moves part of the verifier's work into preprocessing and an additional proof. Table 5 gives all five measured input sizes; Figure 9 shows the full scaling curves.

Table 5: Lattice PCS measurements; Figure 9 plots the same range. Proof size excludes the commitment; RAM is process peak RSS. RoKoKo uses its native instances as described above; a dash denotes an unsupported input.

	Akita						Baselines					
Bits committed		Commit (s)	Open (s)	Verify (ms)	Proof (KB)	RAM (GiB)		Commit (s)	Open (s)	Verify (ms)	Proof (KB)	RAM (GiB)
2^{27}	direct	0.039	0.510	6.1	57.7	0.108	Greyhound	0.111	0.187	77.4	60.1	0.33
	offloaded	—	—	—	—	—	RoKoKo	0.038	0.144	4.1	109.3	0.85
2^{29}	direct	0.141	0.704	7.2	58.2	0.223	Greyhound	0.442	0.393	148	60.6	1.08
	offloaded	0.139	1.07	6.7	62.6	0.232	RoKoKo	0.129	0.246	3.9	109.4	1.7
2^{31}	direct	0.536	1.35	11.5	59.5	0.627	Greyhound	2.10	1.04	329	63.3	3.93
	offloaded	0.534	1.86	9.9	62.7	0.671	RoKoKo	0.471	0.668	4.6	114.8	4.47
2^{33}	direct	2.07	2.32	18.1	60.9	1.21	Greyhound	11.6	4.60	715	64.9	20.7
	offloaded	2.22	3.34	11.4	63.3	1.33	RoKoKo	1.83	1.63	4.3	114.8	12.3
2^{35}	direct	8.42	5.69	32.2	61.0	4.37	Greyhound	54.1	20.1	1537	67.2	92.6
	offloaded	10.7	8.35	14.1	63.7	4.47	RoKoKo	8.90	4.70	7.2	115.0	36.9

Calibrating Greyhound. To align the concrete security targets, we make three changes to Greyhound's reference implementation (Appendix N). We recalibrate the inner and outer commitment ranks for every matrix role to meet a 128-bit quantum ADPS16 target under Euclidean SIS collision bounds. We also correct the Johnson–Lindenstrauss norm check: the original dense-sign projection does not match the distribution required by the certified lower-tail bound. We replace it with a sparse-ternary projection and use the corresponding norm slack $\sqrt{128/29}$ in place of 2. Finally, to handle completeness errors when an honest folded response exceeds the calibrated norm bounds, we add transcript-bound challenge grinding at every folding level, including the root and terminal levels. This lets the prover retry the challenges without changing the preceding commitments. These changes improve comparability but do not remove the distinction between Greyhound's truncated-output assumption and Akita's standard Module-SIS assumption.

Verification and communication. Akita's main advantage over Greyhound is verification time. Across the supported offloaded instances, verification takes 6.7–14.1 ms, compared with 148–1537 ms for Greyhound. The corresponding speedup grows from 22.1× to 108.7×. Akita's opening proofs remain 57.7–63.7 KB across the direct and offloaded configurations, compared with 60.1–67.2 KB for Greyhound; Akita's commitments are 128 bytes. RoKoKo verifies faster, in 3.9–7.2 ms, but targets only 100-bit statistical soundness and has 109.3–115.0 KB proofs (Table 5).

The direct/offloaded comparison shows the price of setup offloading. Offloading is first supported at 2^{29} bits, or 2^{24} coefficients; the 2^{27}-bit instance uses direct evaluation only. At the largest input, offloading reduces verification from 32.2 to 14.1 ms, while opening time grows from 5.69 to 8.35 seconds and proof size from 61.0 to 63.7 KB. Commitment time changes by less than 2% through 2^{31} bits. At 2^{33} and 2^{35} bits the planner selects a different root fold for the offloaded schedule, and commitment time rises from 2.07 to 2.22 seconds and from 8.42 to 10.7 seconds. Thus the verifier improvement costs proof size and prover time (Table 5).

At 2^{35} bits over the 32-bit field, most of the added prover time comes from this root fold rather than from opening the setup. A single-threaded profile of this instance on a separate machine attributes about one third of the added opening time to fold level 1, whose sum-checks also run over the precommitted setup prefix. That prefix has 2^{23} coefficients against 2^{30} in the input, so its share shrinks as the input grows. The rest of the added opening time, and all of the added commitment time, come from the root parameters. With offloading enabled, the planner first minimizes the largest setup prefix left to the verifier (Section 12). The root fold it selects splits each input coefficient into five base-2^7 digits instead of four base-2^8 digits, and draws fold challenges from a 64-dimensional subring instead of a 128-dimensional one, with ten coefficients

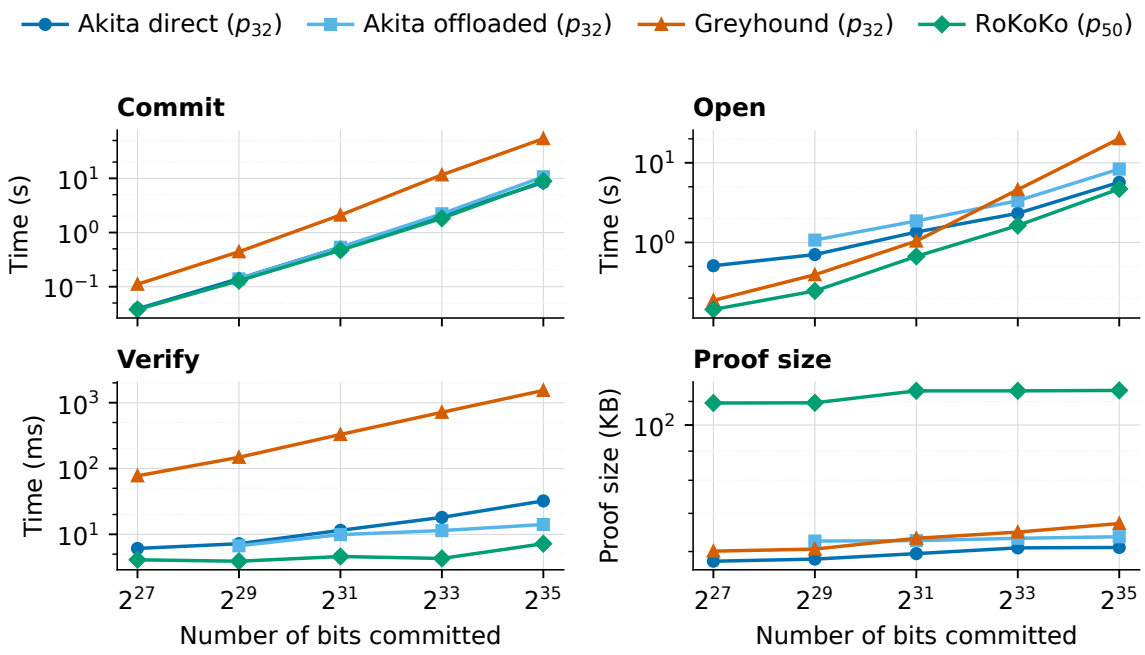


Fig. 9: Lattice PCS comparison. Input size is the nominal number of bits committed, $B = N \log_2 |\mathbb{F}|$; RoKoKo uses its closest supported native instance. Proof size excludes the commitment and evaluation. Missing points denote unsupported inputs.

in $\{\pm 2\}$ in addition to the 31 in $\{\pm 1\}$. The fifth digit widens the root commitment matrix by 25%. The root response takes one sparse multiply-accumulate per digit and nonzero challenge coefficient, $5 \cdot 41$ instead of $4 \cdot 31$, a factor of about 1.65 on the largest single step of the opening. At the same size, offloading changes the opening time from 4.90 to 5.46 seconds over the 64-bit field and from 6.35 to 6.21 seconds over the 128-bit field.

Commitment, opening, and memory. The split between commitment and opening is important. At the largest input, direct Akita commits in 8.42 seconds and opens in 5.69 seconds, compared with 54.1 and 20.1 seconds for Greyhound. Through 2^{31} bits, Greyhound opens faster even though Akita commits faster. RoKoKo has the fastest opening prover at each of its supported sizes. Akita uses the least peak memory in the measured lattice workloads: at the largest input, direct and offloaded Akita use 4.37 and 4.47 GiB, compared with 92.6 GiB for Greyhound and 36.9 GiB for RoKoKo. These are full-process measurements, including storage for the polynomial (Table 5).

13.3 Comparison with Hash-Based PCSs

We compare Akita with FRI [16], STIR [8], WHIR [9], BaseFold [93], and Ligerito [85] implementations from Plonky2, Plonky3, Binius64, Flock, WorldFnd, and SP1. The comparison includes the three Akita prime-field profiles, each in its direct and offloaded configuration. The measurements compare the implementations under their native interfaces and parameter choices. Appendix M records the variable counts, omitted packing reductions, and security configurations. Plonky3 FRI/STIR measure batched univariate openings, with 128 polynomials at 2^{35} bits, rather than the single multilinear opening measured by Akita (Appendix M).

Figure 10 shows both prover thread counts at 2^{35} bits committed, where setup offloading materially reduces Akita's verifier time. Table 6 gives eight-thread prover measurements at 2^{27}, 2^{31}, and 2^{35} bits.

The measurements show a proof-size/runtime tradeoff rather than one implementation winning every metric. At 2^{35} bits, Akita's proofs occupy 61.0–67.9 KB, while the hash-based proofs measured at this size range from 401.6 to 666.7 KB. Akita commits in 4.59–10.7 seconds with one thread and 1.24–1.75 seconds with eight threads; the measured hash baselines take 10.7–17.2 and 3.02–6.27 seconds. Binius64

Table 6: Hash PCS measurements with eight prover threads. Akita subscripts give the prime-field bit width; rows marked (1) use batched univariate FRI/STIR at the two larger sizes (Appendix M). Blank: configuration not included at that size; n/m: not measured.

	Commit (s)			Open (s)			Verify (ms)			Proof (KB)			RAM (GiB)		
Scheme bits:	2^{27}	2^{31}	2^{35}	2^{27}	2^{31}	2^{35}	2^{27}	2^{31}	2^{35}	2^{27}	2^{31}	2^{35}	2^{27}	2^{31}	2^{35}
Akita (p_{32})	0.00827	0.0748	1.24	0.143	0.275	0.946	6.0	11.7	32.2	57.7	59.5	61.0	0.147	0.663	4.4
Akita offload (p_{32})		0.0736	1.75		0.364	1.33		10.2	14.1		62.7	63.7		0.696	4.5
Akita (p_{64})	0.00817	0.0499	1.3	0.0966	0.191	0.823	3.9	7.3	18.0	59.4	62.4	63.0	0.133	0.454	4.61
Akita offload (p_{64})		0.0463	1.27		0.24	0.997		5.9	10.1		64.7	66.8		0.519	4.77
Akita (p_{128})	0.00977	0.0742	1.43	0.0789	0.188	0.951	4.2	9.1	22.2	61.7	64.3	65.0	0.153	0.507	4.81
Akita offload (p_{128})			1.41			1.01			10.7			67.9			4.98
Plonky2 FRI	0.548	9.5	*n/m*	1.2	20.3	*n/m*	14.5	19.5	*n/m*	637.2	846.0	*n/m*	0.957	15.3	*n/m*
Plonky3 FRI(1)	0.205	0.686	6.27	0.241	0.46	0.687	7.1	7.8	8.5	410.6	452.7	532.7	1.17	2.78	12.2
Plonky3 STIR(1)	0.21	0.687	6.27	0.43	0.857	1.09	5.8	6.1	6.9	290.9	312.4	401.6	1.17	2.78	12.2
WHIR (Plonky3)	0.0205	0.445	5.23	0.146	3.04	12.7	3.4	4.4	6.1	211.4	279.9	410.4	0.24	3.7	18.9
Binius64 BaseFold	0.00664	0.23	5.43	0.00725	0.0923	1.31	0.4	0.6	0.9	308.6	472.8	666.7	0.0913	1.42	22.7
Flock Ligerito	0.0094	0.165	3.02	0.0185	0.265	4.46	1.2	1.2	1.5	408.9	494.6	570.5	0.0905	1.35	21.6
WHIR (WorldFnd)	0.0673	1.37	*n/m*	0.348	6.3	*n/m*	1.0	1.2	*n/m*	196.6	256.9	*n/m*	0.181	2.85	*n/m*
BaseFold (SP1)	2.17	5.39	*n/m*	3.25	3.32	*n/m*	25.0	25.4	*n/m*	1179.4	1194.8	*n/m*	1.33	2.5	*n/m*

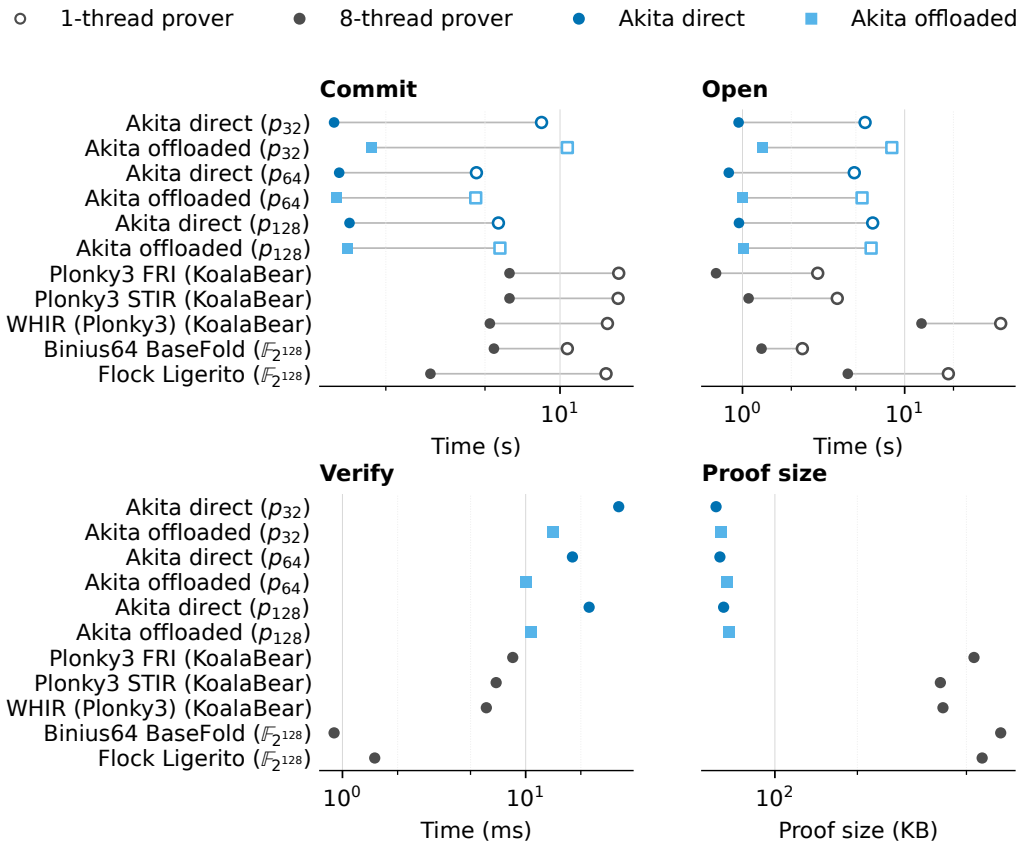


Fig. 10: Hash PCS comparison at 2^{35} bits committed. Commitment and opening show one- and eight-thread proving. Proof size excludes the commitment. Plonky3 FRI and STIR use batched univariate instances. Baselines without a completed measurement at this size are omitted.

Table 7: Jolt measurements at every other padded trace length T on the M4 Max. Times and peak RAM are medians of three runs; proof size is the complete Jolt proof. Akita first uses setup offloading at $T = 2^{21}$; Jolt-with-Dory widens its one-hot chunks at $T = 2^{25}$.

	Prove (s)		Verify (ms)		Proof (KB)		RAM (GiB)	
T	Akita	Dory	Akita	Dory	Akita	Dory	Akita	Dory
2^{17}	0.33	1.16	8.29	67.51	81.8	81.9	0.30	0.24
2^{19}	0.59	2.38	11.50	71.04	84.4	86.2	0.45	0.48
2^{21}	1.73	5.41	13.28	75.26	88.7	90.6	1.06	1.20
2^{23}	5.55	14.78	19.30	79.46	90.0	94.9	1.97	2.98
2^{25}	20.72	54.31	23.33	82.16	91.5	91.8	6.43	9.71
2^{27}	75.05	171.93	24.88	86.19	93.6	95.9	24.37	27.49

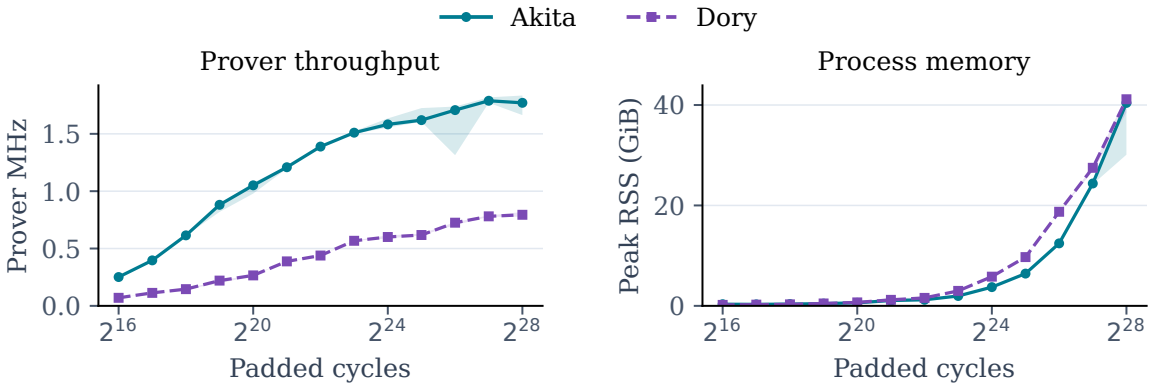


Fig. 11: SHA-256 chains on the M4 Max. Left: padded prover throughput. Right: process peak RSS. Points show medians of three runs; bands show observed ranges.

BaseFold, Plonky3 FRI, and Plonky3 STIR open faster than all six Akita configurations in the one-thread measurements, while Flock Ligerito and Plonky3 WHIR open more slowly. All five also verify faster than all six Akita configurations (Figure 10 and Table 6). These are workload-specific tradeoffs: the field and packing choices change the opening problem as well as the implementation cost.

13.4 Jolt End-to-End

To test whether Akita's PCS tradeoffs benefit a complete prover, we compare Jolt-with-Akita and Jolt-with-Dory on SHA-256 chains with padded trace lengths $T = 2^{16}, 2^{17}, \ldots, 2^{28}$. Both configurations use Jolt's optimized CPU backend on an Apple M4 Max with 16 CPU cores and 64 GiB RAM, running macOS 27.0.1. We run each configuration three times at every trace length, alternating the order of the two configurations; all 78 proofs verify successfully. The figures show medians and the full observed ranges.

Prover time includes witness materialization, commitments, sum-checks, and the joint opening proof; guest compilation and execution tracing, preprocessing, and PCS setup are excluded. We report throughput in millions of padded trace rows per second (MHz), and memory as process peak RSS. Verification times use a single worker. Proof sizes are the complete serialized Jolt proofs. Revisions and per-run measurements, including Perfetto traces, are recorded in the reproduction artifact; tracing is enabled throughout.

Table 7 gives trace lengths $2^{17}, 2^{19}, \ldots, 2^{27}$; the graphs retain all thirteen measured lengths.

Proving and memory. Across these trace lengths, Jolt-with-Akita proves 2.2–4.2× faster than Jolt-with-Dory (Figure 11). At $T = 2^{27}$, the median prover time falls from 171.93 to 75.05 seconds, increasing padded throughput from 0.781 to 1.788 MHz. Akita uses less memory from $T = 2^{19}$ upward: at $T = 2^{26}$, median peak RSS is 12.45 GiB for Akita and 18.74 GiB for Dory, and at $T = 2^{27}$ it is 24.37 and 27.49 GiB, respectively. Below 2^{19}, where both provers use under 0.4 GiB, Akita uses slightly more. At $T = 2^{28}$, the largest trace length, peak RSS is 40.42 GiB for Akita and 41.15 GiB for Dory, and throughput is 1.771 and 0.794 MHz.

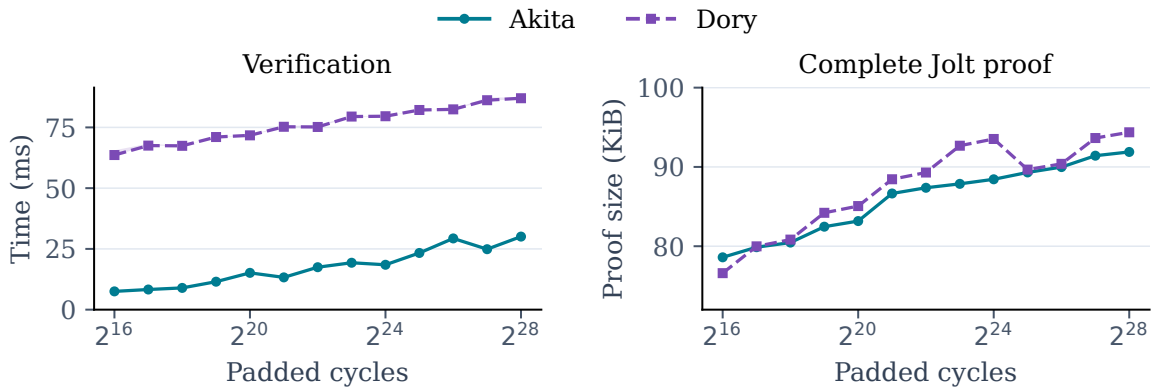


Fig. 12: SHA-256 chains on the M4 Max. Left: verification time, showing medians and observed ranges over three runs. Right: complete proof size in KiB.

Verification and proof size. Akita reduces verification time by 2.8–8.5× (Figure 12). At $T = 2^{27}$, verification takes 24.88 ms with Akita and 86.19 ms with Dory. The faster prover and verifier do not require more communication: at each trace length, Akita's complete proof is between 5.4% smaller and 2.6% larger than Dory's, and no proof exceeds 100 KB (the largest, Dory's at $T = 2^{28}$, is 96,634 bytes). Dory's proof size is identical across repetitions; Akita's is identical up to $T = 2^{25}$ and varies by at most 117 bytes from $T = 2^{26}$ upward.

Configuration transitions. The curves are not smooth because the implementation changes configurations at two trace lengths. At $T = 2^{21}$, Akita first uses setup offloading for its one-hot commitments. Verification falls from 15.15 to 13.28 ms despite the larger trace, while proof size grows from 85.2 to 88.7 KB. At $T = 2^{25}$, Jolt-with-Dory switches from 4-bit to 8-bit committed one-hot chunks, whereas Jolt-with-Akita keeps 4-bit chunks at every trace length. Wider chunks reduce the number of committed polynomials: Dory's proof shrinks from 95.8 to 91.8 KB, while Akita's grows from 90.6 to 91.5 KB. Akita's verification rises from 18.47 to 23.33 ms between these lengths, and its prover throughput continues to increase (Figures 11 and 12).

14 Conclusion and Future Work

We end with several directions for future work and applications of Akita. Our first target is to reduce the proof size further, through both cheaper folds and a smaller tail; our second is to improve proving time.

We also want to strengthen both the completeness and the soundness arguments. On the completeness side, the response-model premise used for optimized later-fold schedules is heuristic. We would like to prove it rigorously or, failing that, to quantify the precise cost of switching to rigorously proven bounds (which can still be average-case rather than worst-case). On the soundness side, the main gap is the lack of a QROM analysis for Akita; the literature shows promising progress, but these results do not yet seem to transfer fully to Akita.

As for applications, we see Jolt as only the starting point for Akita. Akita facilitates better proof systems for natively lattice-based claims, such as aggregating the post-quantum lattice signatures ML-DSA [79] and Falcon (FN-DSA) [44]. Their verification equations consist of structured lattice relations and norm bounds, and prior work already expresses Falcon aggregation through LaBRADOR relations [1,18]. A broader and more ambitious target is verifiable FHE [92].

Finally, we would like to bring zero-knowledge to Akita, ideally with low concrete overhead in proof size, prover time, and verifier time. This would unlock privacy applications such as proofs of possession for signatures and verifiable pseudorandom functions.

Acknowledgments

We thank Michael Zhu, Andrew Tretyakov, and Alberto Centelles for helpful discussions and assistance integrating Akita into Jolt.

Disclosures

Justin Thaler is a Research Partner at a16z crypto and is an investor in various blockchain-based platforms, as well as in the crypto ecosystem more broadly (for general a16z disclosures, see https://www.a16z.com/disclosures/).

AI Assistance

Author A used GPT 5.5, GPT 5.6 Sol, GPT 6 Astra, GPT 6.1 Sol, and Claude Opus 5.5 extensively to draft, restructure, and review this manuscript, directing revisions through detailed instructions, corrections, and selective approval. AI tools also assisted the Rust implementation and benchmark infrastructure. The authors retain responsibility for the paper's claims, proofs, experimental results, and references, including all AI-assisted material.

References

1. Aardal, M.A., Aranha, D.F., Boudgoust, K., Kolby, S., Takahashi, A.: Aggregating falcon signatures with LaBRADOR. Cryptology ePrint Archive, Paper 2024/311 (2024), https://eprint.iacr.org/2024/311
2. Ajtai, M.: Generating hard instances of lattice problems. In: Proc. 28th Annual ACM Symp. on Theory of Computing (STOC). pp. 99–108. ACM (1996). https://doi.org/10.1145/237814.237838, https://doi.org/10.1145/237814.237838
3. Albrecht, M.R., Fenzi, G., Lapiha, O., Nguyen, N.K.: SLAP: Succinct lattice-based polynomial commitments from standard assumptions. Cryptology ePrint Archive, Paper 2023/1469 (2023), https://eprint.iacr.org/2023/1469, EUROCRYPT 2024
4. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of Learning with Errors. Journal of Mathematical Cryptology **9**(3), 169–203 (Oct 2015). https://doi.org/10.1515/jmc-2015-0016, extended version: https://eprint.iacr.org/2015/046. The `lattice-estimator` Sage package implements estimates from this framework.
5. Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: Post-quantum key exchange—a new hope. In: 25th USENIX Security Symposium (USENIX Security 16). pp. 327–343. USENIX Association (2016), https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/alkim
6. Ames, S., Hazay, C., Ishai, Y., Venkitasubramaniam, M.: Ligero: Lightweight sublinear arguments without a trusted setup. In: 2017 ACM SIGSAC Conference on Computer and Communications Security (2017)
7. Arnon, G., Boneh, D., Fenzi, G.: Open problems in list decoding and correlated agreement. Cryptology ePrint Archive, Paper 2026/680 (2026), https://eprint.iacr.org/2026/680
8. Arnon, G., Chiesa, A., Fenzi, G., Yogev, E.: STIR: Reed–solomon proximity testing with fewer queries. Cryptology ePrint Archive, Paper 2024/390 (2024), https://eprint.iacr.org/2024/390
9. Arnon, G., Chiesa, A., Fenzi, G., Yogev, E.: WHIR: Reed–solomon proximity testing with super-fast verification. Cryptology ePrint Archive, Paper 2024/1586 (2024), https://eprint.iacr.org/2024/1586
10. Arun, A., Setty, S., Thaler, J.: Jolt: Snarks for virtual machines via lookups. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 3–33. Springer (2024)
11. Arun, A., Setty, S., Thaler, J.: Jolt codebase. https://github.com/a16z/jolt (2025)
12. Attema, T., Fehr, S., Klooß, M.: Fiat-shamir transformation of multi-round interactive proofs. In: Theory of Cryptography Conference. pp. 113–142. Springer (2022)
13. Attema, T., Lyubashevsky, V., Seiler, G.: Practical product proofs for lattice commitments. In: Micciancio, D., Ristenpart, T. (eds.) Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17–21, 2020, Proceedings, Part II. Lecture Notes in Computer Science, vol. 12171, pp. 470–499. Springer (2020). https://doi.org/10.1007/978-3-030-56880-1_17, https://eprint.iacr.org/2020/517
14. Balbás, D., Nitulescu, A., Plançon, M.: ProtogaLattice: Constant-round lattice-based folding for general polynomial relations. Cryptology ePrint Archive, Paper 2026/1317 (2026), https://eprint.iacr.org/2026/1317

15. Becker, A., Ducas, L., Gama, N., Laarhoven, T.: New directions in nearest neighbor searching with applications to lattice sieving. In: Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms. pp. 10–24. SIAM (2016). https://doi.org/10.1137/1.9781611974331.ch2, https://doi.org/10.1137/1.9781611974331.ch2
16. Ben-Sasson, E., Bentov, I., Horesh, Y., Riabzev, M.: Fast Reed-Solomon interactive oracle proofs of proximity. In: ICALP (2018)
17. Ben-Sasson, E., Carmon, D., Ishai, Y., Kopparty, S., Saraf, S.: Proximity gaps for Reed-Solomon codes (2020)
18. Beullens, W., Seiler, G.: Labrador: Compact proofs for R1CS from module-sis. In: Handschuh, H., Lysyanskaya, A. (eds.) Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part V. pp. 518–548. Lecture Notes in Computer Science, Springer (2023). https://doi.org/10.1007/978-3-031-38554-4_17, https://doi.org/10.1007/978-3-031-38554-4_17
19. Biasioli, B., Bolboceanu, M., Lyubashevsky, V., Merino-Gallardo, A., Osadnik, M., Seiler, G., Steuer, P.: A toolkit for succinct lattice-based zero knowledge proofs. Cryptology ePrint Archive, Paper 2026/1289 (2026), https://eprint.iacr.org/2026/1289
20. Bolboceanu, M., Bootle, J., Lyubashevsky, V., Merino-Gallardo, A., Seiler, G.: Orthus: Practical sublinear batch-verification of lattice relations from standard assumptions. Cryptology ePrint Archive, Paper 2026/398 (2026), https://eprint.iacr.org/2026/398, CRYPTO 2026
21. Boneh, D., Chen, B.: LatticeFold: A lattice-based folding scheme and its applications to succinct proof systems. Cryptology ePrint Archive, Paper 2024/257 (2024), https://eprint.iacr.org/2024/257
22. Boneh, D., Chen, B.: Latticefold+: Faster, simpler, shorter lattice-based folding for succinct proof systems. Cryptology ePrint Archive (2025)
23. Bonnetain, X., Chailloux, A., Schrottenloher, A., Shen, Y.: Finding many collisions via reusable quantum walks. Cryptology ePrint Archive, Paper 2022/676 (2022), https://eprint.iacr.org/2022/676, EUROCRYPT 2023
24. Bootle, J., Chiesa, A., Sotiraki, K.: Sumcheck arguments and their applications. Cryptology ePrint Archive, Report 2021/333 (2021)
25. Bootle, J., Chiesa, A., Sotiraki, K.: Lattice-based succinct arguments for NP with polylogarithmic-time verification. Cryptology ePrint Archive, Paper 2023/930 (2023), https://eprint.iacr.org/2023/930, CRYPTO 2023
26. Bootle, J., Lyubashevsky, V., Nguyen, N.K., Seiler, G.: A non-PCP approach to succinct quantum-safe zero knowledge. In: CRYPTO (2020)
27. Brehm, M., Chen, B., Fisch, B., Resch, N., Rothblum, R.D., Zeilberger, H.: Blaze: Fast snarks from interleaved raa codes. Cryptology ePrint Archive (2024)
28. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short proofs for confidential transactions and more. In: 2018 IEEE Symposium on Security and Privacy (2018)
29. Bünz, B., Fisch, B., Szepieniec, A.: Transparent SNARKs from DARK compilers. In: Advances in Cryptology – EUROCRYPT 2020 (2020)
30. Canetti, R., Chen, Y., Holmgren, J., Lombardi, A., Rothblum, G.N., Rothblum, R.D.: Fiat-shamir from simpler assumptions. Cryptology ePrint Archive (2018)
31. Chen, B.: Symphony: Toward optimal zero-knowledge snarks from lattices. Cryptology ePrint Archive, Paper 2025/1905 (2025), https://eprint.iacr.org/2025/1905
32. Chiesa, A., Hu, Y., Maller, M., Mishra, P., Vesely, N., Ward, N.: Marlin: Preprocessing zkSNARKs with universal and updatable SRS. In: Advances in Cryptology – EUROCRYPT 2020 (2020)
33. Chiesa, A., Orrù, M.: A Fiat–Shamir transformation from duplex sponges. Cryptology ePrint Archive, Paper 2025/536 (2025), https://eprint.iacr.org/2025/536
34. Cini, V., Lai, R.W.F., Malavolta, G.: Lattice-based succinct arguments from vanishing polynomials. In: Handschuh, H., Lysyanskaya, A. (eds.) Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part II. Lecture Notes in Computer Science, vol. 14082, pp. 72–105. Springer (2023). https://doi.org/10.1007/978-3-031-38545-2_3, https://doi.org/10.1007/978-3-031-38545-2_3
35. Cini, V., Malavolta, G., Nguyen, N.K., Wee, H.: Polynomial commitments from lattices: Post-quantum security, fast verification and transparent setup. Cryptology ePrint Archive, Paper 2024/281 (2024), https://eprint.iacr.org/2024/281, CRYPTO 2024
36. Crosswhite, G.M., Bacon, D.: Finite automata for caching in matrix product algorithms. Physical Review A **78**(1), 012356 (2008). https://doi.org/10.1103/PhysRevA.78.012356, https://arxiv.org/abs/0708.1221
37. Dao, Q., DeStefano, Z., Bagad, S., Domb, Y., Thaler, J.: Speeding up sum-check proving (extended version). Cryptology ePrint Archive, Paper 2026/587 (2026), https://eprint.iacr.org/2026/587
38. Diamond, B.E., Posen, J.: Succinct arguments over towers of binary fields. Cryptology ePrint Archive, Paper 2023/1784 (2023), https://eprint.iacr.org/2023/1784, https://eprint.iacr.org/2023/1784

39. Diamond, B.E., Posen, J.: Polylogarithmic proofs for multilinears over binary towers. Cryptology ePrint Archive, Paper 2024/504 (2024), https://eprint.iacr.org/2024/504, https://eprint.iacr.org/2024/504
40. Ethereum Foundation: Ethproofs: Tracking real-time proving for ethereum. https://ethproofs.org/ (2025)
41. Ethereum Foundation: Shipping an L1 zkevm #1: Realtime proving. https://blog.ethereum.org/2025/07/10/realtime-proving (2025)
42. Ethereum Foundation: The proximity prize. https://proximityprize.org/ (2026)
43. Fenzi, G., Moghaddas, H., Nguyen, N.K.: Lattice-based polynomial commitments: Towards asymptotic and concrete efficiency. Journal of Cryptology **37**(3), 31 (2024). https://doi.org/10.1007/s00145-024-09511-8, https://doi.org/10.1007/s00145-024-09511-8
44. Fouque, P., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Prest, T., Ricosset, T., Seiler, G., Whyte, W., Zhang, Z.: Falcon: Fast-fourier lattice-based compact signatures over NTRU. NIST Post-Quantum Cryptography Round-3 submission, Specification v1.2 (2020), https://falcon-sign.info/falcon.pdf
45. Gabizon, A., Williamson, Z.J., Ciobotaru, O.: PLONK: Permutations over Lagrange-bases for oecumenical non-interactive arguments of knowledge. ePrint Report 2019/953 (2019)
46. Garreta, A., Lipmaa, H., Luhaäär, U., Osadnik, M.: Cyclo: Lightweight lattice-based folding via partial range checks. Cryptology ePrint Archive, Paper 2026/359 (2026), https://eprint.iacr.org/2026/359
47. Goldwasser, S., Kalai, Y.T., Rothblum, G.N.: Delegating computation: interactive proofs for muggles. Journal of the ACM (JACM) **62**(4), 1–64 (2015)
48. Golovnev, A., Lee, J., Setty, S., Thaler, J., Wahby, R.S.: Brakedown: Linear-time and post-quantum snarks for R1CS. Cryptology ePrint Archive (2021)
49. Groth, J.: On the size of pairing-based non-interactive arguments. In: Advances in Cryptology – EUROCRYPT 2016 (2016)
50. Gruen, A.: Some improvements for the PIOP for ZeroCheck. Cryptology ePrint Archive, Paper 2024/108 (2024), https://eprint.iacr.org/2024/108
51. Gürkan, K., Novakovic, A., Rothblum, R.D.: Bolt: Faster SNARKs from sketched codes. Cryptology ePrint Archive, Paper 2026/310 (2026), https://eprint.iacr.org/2026/310
52. Hemo, T., Jue, K., Rabinovich, E., Roh, G., Rothblum, R.D.: Jagged polynomial commitments (or: How to stack multilinears). Cryptology ePrint Archive, Paper 2025/917 (2025), https://eprint.iacr.org/2025/917
53. Holmgren, J., Rothblum, R.D.: Delegating computations with (almost) minimal time and space overhead. In: 59th IEEE Annual Symposium on Foundations of Computer Science. pp. 124–135 (2018). https://doi.org/10.1109/FOCS.2018.00021, https://ieee-focs.org/FOCS-2018-Papers/pdfs/59f124.pdf
54. Huang, M.M., Mao, X., Zhang, J.: Sublinear proofs over polynomial rings. Cryptology ePrint Archive, Paper 2025/199 (2025), https://eprint.iacr.org/2025/199
55. Hwang, I., Lee, H., Seo, J., Song, Y.: Jindo: Practical lattice-based polynomial commitments for client-side proving. Cryptology ePrint Archive, Paper 2026/044 (2026), https://eprint.iacr.org/2026/044
56. Hwang, I., Seo, J., Song, Y.: Concretely efficient lattice-based polynomial commitment from standard assumptions. Cryptology ePrint Archive, Paper 2024/306 (2024), https://eprint.iacr.org/2024/306
57. Kallesøe, A., Khoshakhlagh, H.: Grand Danois: Succinct multilinear polynomial commitments over lattices. Cryptology ePrint Archive, Paper 2026/1196 (2026), https://eprint.iacr.org/2026/1196, revised August 21, 2026
58. Kallesøe, A., Khoshakhlagh, H.: Grand Danois: Succinct multilinear polynomial commitments over lattices. Cryptology ePrint Archive, Paper 2026/1196, first public version (2026), https://eprint.iacr.org/archive/2026/1196/1780873625.pdf, version dated June 7, 2026
59. Kate, A., Zaverucha, G.M., Goldberg, I.: Constant-size commitments to polynomials and their applications. In: Advances in Cryptology – ASIACRYPT 2010. pp. 177–194 (2010)
60. Klooß, M., Lai, R.W.F., Nguyen, N.K., Osadnik, M.: RoK, paper, SISsors – toolkit for lattice-based succinct arguments. Cryptology ePrint Archive, Paper 2024/1972 (2024), https://eprint.iacr.org/2024/1972
61. Klooß, M., Lai, R.W.F., Nguyen, N.K., Osadnik, M.: Rok and roll - verifier-efficient random projection for o˜(λ)-size lattice arguments - (extended abstract). In: Hanaoka, G., Yang, B. (eds.) Advances in Cryptology - ASIACRYPT 2025 - 31st International Conference on the Theory and Application of Cryptology and Information Security, Melbourne, VIC, Australia, December 8-12, 2025, Proceedings, Part III. pp. 297–329. Lecture Notes in Computer Science, Springer (2025). https://doi.org/10.1007/978-981-95-5099-9_10, https://doi.org/10.1007/978-981-95-5099-9_10
62. Klooß, M., Lai, R.W.F., Nguyen, N.K., Osadnik, M., Tucci, L.: RoKoKo: Lattice-based succinct arguments, a committed refinement. Cryptology ePrint Archive, Paper 2026/575 (2026), https://eprint.iacr.org/2026/575, revised September 9, 2026
63. Kuriyama, S., Lai, R.W.F., Osadnik, M., Tucci, L.: SALSAA – sumcheck-aided lattice-based succinct arguments and applications. Cryptology ePrint Archive, Paper 2025/2124 (2025), https://eprint.iacr.org/2025/2124
64. Langlois, A., Stehlé, D.: Worst-case to average-case reductions for module lattices. Cryptology ePrint Archive, Paper 2012/090 (2012), https://eprint.iacr.org/2012/090

65. LayerZero Labs: The akita book. https://github.com/LayerZero-Labs/akita/tree/main/book (2026)
66. LayerZero Labs: Akita codebase. https://github.com/LayerZero-Labs/akita (2026)
67. LayerZero Labs: Akita v0.1.0. https://github.com/LayerZero-Labs/akita/tree/029ceceaba879a0d9bf80a47d2e08534e630457d (2026), release revision inspected September 11, 2026
68. LayerZero Labs: Greyhound reference: Parameter selection and recursive packing. Source-code artifact (2026), https://github.com/LayerZero-Labs/greyhound-reference/tree/687a6f8be1dbc5bf1fa3927bb4a0a8d1e84d8397, inspected revision 687a6f8, files greyhound.c, labrador.c, and pack.c
69. LayerZero Labs: PCS Benchmarks. https://github.com/LayerZero-Labs/pcs-benchmarks (2026)
70. LaZer authors: LaBRADOR: Parameter generation and proof verification. Source-code artifact (2026), https://github.com/lazer-crypto/labrador/tree/3f95485139ffaa65fe572da809b90772901372e5, inspected revision 3f95485, files labrador_core.c and pack.c
71. leanEthereum: leanVM: Minimal hash-based zkvm for a post-quantum ethereum. https://github.com/leanEthereum/leanVM (2026)
72. Lee, J.: Dory: Efficient, transparent arguments for generalised inner products and polynomial commitments. In: Theory of Cryptography Conference. pp. 1–34. Springer (2021)
73. Lighter Team: Lighter Core: Technical Architecture and Prover Implementation. https://docs.lighter.xyz/about-lighter/technical-architecture-lighter-core (2026), public prover implementation: https://github.com/elliottech/lighter-prover
74. Lund, C., Fortnow, L., Karloff, H., Nisan, N.: Algebraic methods for interactive proof systems. In: FOCS (Oct 1990)
75. Lyubashevsky, V.: Fiat-shamir with aborts: Applications to lattice and factoring-based signatures. In: Matsui, M. (ed.) Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009, Proceedings. Lecture Notes in Computer Science, vol. 5912, pp. 598–616. Springer (2009). https://doi.org/10.1007/978-3-642-10366-7_35, https://doi.org/10.1007/978-3-642-10366-7_35
76. Lyubashevsky, V., Nguyen, N.K., Plançon, M.: Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part II. Lecture Notes in Computer Science, vol. 13508, pp. 71–101. Springer (2022). https://doi.org/10.1007/978-3-031-15979-4_3, https://doi.org/10.1007/978-3-031-15979-4_3
77. Lyubashevsky, V., Seiler, G.: Short, invertible elements in partially splitting cyclotomic rings and applications to lattice-based zero-knowledge proofs. In: Nielsen, J.B., Rijmen, V. (eds.) Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part I. pp. 204–224. Lecture Notes in Computer Science, Springer (2018). https://doi.org/10.1007/978-3-319-78381-9_8, https://doi.org/10.1007/978-3-319-78381-9_8
78. MATZOV: Report on the security of LWE: Improved dual lattice attack. Technical report (2022). https://doi.org/10.5281/zenodo.6412487, https://doi.org/10.5281/zenodo.6412487
79. National Institute of Standards and Technology: FIPS 204: Module-lattice-based digital signature standard. Federal Information Processing Standards Publication (2024), https://doi.org/10.6028/NIST.FIPS.204
80. Nguyen, N.K., O'Rourke, G., Zhang, J.: Hachi: Efficient lattice-based multilinear polynomial commitments over extension fields. IACR Cryptol. ePrint Arch. **2026**, 156 (2026), https://eprint.iacr.org/2026/156
81. Nguyen, N.K., O'Rourke, G., Zhang, J.: Personal communication concerning the base-field optimization in Hachi (2026)
82. Nguyen, N.K., Seiler, G.: Greyhound: Fast polynomial commitments from lattices. In: Reyzin, L., Stebila, D. (eds.) Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part X. pp. 243–275. Lecture Notes in Computer Science, Springer (2024). https://doi.org/10.1007/978-3-031-68403-6_8, https://doi.org/10.1007/978-3-031-68403-6_8
83. Nguyen, W., Setty, S.: Neo: Lattice-based folding scheme for ccs over small fields and pay-per-bit commitments. Cryptology ePrint Archive (2025)
84. Nguyen, W., Setty, S.: Neo and SuperNeo: Post-quantum folding with pay-per-bit costs over small fields. Cryptology ePrint Archive, Paper 2026/242 (2026), https://eprint.iacr.org/2026/242
85. Novakovic, A., Angeris, G.: Ligerito: A small and concretely fast polynomial commitment scheme. Cryptology ePrint Archive, Paper 2025/1187 (2025), https://eprint.iacr.org/2025/1187
86. Osadnik, M.: PikkuFold: Efficient folding in a few kilobytes. Cryptology ePrint Archive, Paper 2026/1809 (2026), https://eprint.iacr.org/2026/1809
87. Raikos, G., Wong, D.: Soundness failures in LaBRADOR implementations from NTT-friendly rings. ZK/SEC Quarterly, zkSecurity (2025), https://blog.zksecurity.xyz/posts/labrador-bugs/
88. RoKoko authors: RoKoko: Concrete parameter configurations. Source-code artifact (2026), https://github.com/lattice-arguments/rokoko/tree/1baa91e901fc37b5fa59e65c26a630cb93849b3e, inspected revision 1baa91e, README and src/protocol/params.rs

89. Royen, T.: A simple proof of the gaussian correlation conjecture extended to some multivariate gamma distributions. Far East Journal of Theoretical Statistics **48**(2), 139–145 (2014)
90. Setty, S., Thaler, J.: Twist and shout: Faster memory checking arguments via one-hot addressing and increments. Cryptology ePrint Archive, Paper 2025/105 (2025), https://eprint.iacr.org/2025/105
91. Succinct: Sp1 hypercube: Proving ethereum in real-time. https://blog.succinct.xyz/sp1-hypercube/ (2025)
92. Wei, Y., Wang, K., Xiang, B., Zhang, X., Wang, H., Deng, Y., Zhu, X., Lin, L.: Packed sumcheck over fields of small characteristic with application to verifiable FHE. Cryptology ePrint Archive, Paper 2025/719 (2025), https://eprint.iacr.org/2025/719
93. Zeilberger, H., Chen, B., Fisch, B.: BaseFold: Efficient field-agnostic polynomial commitment schemes from foldable codes. Cryptology ePrint Archive, Paper 2023/1705 (2023), https://eprint.iacr.org/2023/1705

Table 8: Core algebra, setup, and commitment notation.

Symbol	Meaning
K, E	Base field $K = \mathbb{F}_q$ and opening/challenge field $E = \mathbb{F}_{q^k}$
$R_{q,d}$	Cyclotomic ring $\mathbb{Z}_q[X]/(X^d + 1)$ with dimension d
R, S, S_E	At a coefficient-packing fold, the A ring, challenge subring, and scalar-extended opening ring of (37)
cfg_h	Fold-h configuration $(\mathsf{Open}_h, \mathsf{RingCheck}_h, \mathsf{Comp}_h, \mathsf{Norm}_h)$
Src, P	Frozen source layout and its ordered polynomial count in a commitment shape
edge_h	Outgoing edge tag: direct, setup-offloaded, or terminal inner state
κ_{root}, $N_\star$	Constant-root exponent parameter and target scale $N_\star = \lceil N^{1/\kappa_{\mathsf{root}}} \rceil$
d_A, d_B, d_D	Ring dimensions for the inner-fold, outer-commitment, and partial-evaluation matrices
$d_{F,j}$, $d_{H,j}$	Per-layer ring dimensions of the generated $\mathbf{F}$- and $\mathbf{H}$-chains
$d_{\max,h}$	Largest native ring dimension among the active ring-valued rows of fold h
$\mathbf{A}$, $\mathbf{B}$, $\mathbf{D}$	Public setup matrices for inner folding, outer commitment, and partial evaluation
$(\mathbf{F}_j)_{1\le j\le L_F}$, $(\mathbf{H}_j)_{1\le j\le L_H}$	Two-stage compression matrices (outer and partial-evaluation); $L_F = L_H \in \{0, 2\}$
$\mathcal{M}_{\mathsf{setup}}$	Active $\mathbf{A}/\mathbf{B}/\mathbf{D}$ instances in the fused setup scan and offloaded claim; $\mathbf{F}/\mathbf{H}$ maps are evaluated directly
$\mathbf{S}$, $\mathbf{S}^\flat$	Shared public setup coefficient vector and its canonical zero-padded vector for setup offloading
N_{active}	Length of the longest flat shared-setup prefix read by an active matrix
$n_\bullet$, $m_\bullet$	Row count (module rank) and column count (length) for matrix $\bullet$
N, M	Number of elements in the ring-packed vector and ring elements per block, with $M = 2^{r_{\mathsf{pos}}}$ in the Boolean layout
B, r_{blk}	Block count $B = \lceil N/M \rceil$ and Boolean block-index bits $r_{\mathsf{blk}} = \lceil \log_2 B \rceil$
$\mathbf{s}_i$	Gadget decomposition of block i
$\mathbf{t}_i$	Inner commitment of block i ($\mathbf{A}\mathbf{s}_i \in R_{q,d_A}^{n_A}$)
$\hat{\mathbf{t}}_i$	Digitized inner commitment ($\mathbf{G}_{b_1,n_A}^{-1}(\mathbf{t}_i)$)
L_B, m_B, m_D	Total outer input length, $\mathbf{B}$ slice width, and full unsliced $\mathbf{D}$ width
S_B, $I_{B,s}$	Frozen $\mathbf{B}$ slice count and its derived canonical block range
$\Pi_B(B, S_B)$	Shorthand for $(I_{B,s})_{s\in[S_B]}$, not a separately stored partition
$\mathbf{x}_{B,s}$	Polynomial-major, padded input vector of $\mathbf{B}$ slice s (length m_B)
$\mathbf{u}_s$	Outer image of slice s ($\mathbf{B}\mathbf{x}_{B,s} \in R_{q,d_B}^{n_B}$)
$\mathbf{u}$	Complete outer stack ($(\mathbf{u}_s)_{s\in[S_B]} \in R_{q,d_B}^{S_B n_B}$)
$\mathbf{v}$	One unsliced opening image ($\mathbf{D}\hat{\mathbf{e}} \in R_{q,d_D}^{n_D}$)
$\mathbf{u}_{\mathsf{pub}}$, $\mathbf{v}_{\mathsf{pub}}$	Transmitted outer and opening payloads (raw or terminal compressed images)
C, $C_{S,h}$	An ordinary public commitment payload and the preprocessed setup-prefix commitment selected for offloaded edge h
$(\hat{\mathbf{u}}_j)_{j\in\{1,\dots,L_F\}}$	Digits in the outer compression chain
$(\hat{\mathbf{v}}_j)_{j\in\{1,\dots,L_H\}}$	Digits in the opening compression chain
e_i	Logical partial opening evaluation of block i: an element of the extension-field carrier ring under coefficient packing, or of the A ring under evaluation trace
$\hat{e}_i$	Digitization of the base-field coordinates of e_i

A Notation

Tables 8 and 9 collect the notation used in the protocol description of Section 4 through Section 9.

Table 9: Folding, response certification, and relation-reduction notation.

Symbol	Meaning
κ_h, $\varepsilon_{\mathsf{cw},h}$	Tensor-reduction head arity at receiver h and its distinct coordinate-wise fold error
c_i	Sparse folding challenge for block i: sampled in the challenge subring for coefficient packing and in the A ring for evaluation trace
$\mathbf{z}$	Folded witness $(\sum_i c_i \mathbf{s}_i)$
$\hat{\mathbf{z}}$	Digitized folded witness carried to the next level
$\mathbf{G}_{b_z,m_A}$	Fold-response gadget matrix, with δ_f digits per value
$[-M_f, T_f]$	Exact coefficient interval representable by the honest canonical response digits
$\beta^{\mathsf{cert}}_{\mathsf{fold}}$	Coefficient envelope implied by the response-digit alphabet enforced by the verifier
κ_1, κ_2, Γ	Accepted folding-challenge bounds on coefficient ℓ_1, coefficient ℓ_2, and multiplication-operator norm
$\mathcal{E}_{\mathsf{int}}(\hat{\mathbf{z}})$, $\mathcal{E}_{\mathsf{resp}}$, $S_{\max}$	Squared norm of the reconstructed integer response, its transmitted integer value, and the schedule bound on that value
$p_{\mathsf{acc},h}$, $p^{\mathsf{rsp}}_{h,r}$, $N_{\mathsf{try},h}$	Per-level and per-response admission targets conditional on successful challenge construction, and verifier-enforced nonce allowance
$\underline{\alpha}_{h,u}$, $\underline{a}_h$	Certified accepted fraction for challenge coordinate u and resulting lower bound on complete challenge-construction success at stage h
$\widehat{\delta}^{\mathsf{rsp}}_h$, $\widehat{\delta}^{\mathsf{tot}}_h$	Response-admission failure bound and complete nonce-failure bound including challenge-sampler exhaustion
$p^{\mathsf{box}}_{h,r}$, $p^{\mathsf{ball}}_{h,r}$, $p^{\mathsf{G}}_{h,r}$	Gaussian-surrogate box, ball, and correlated joint response scores
$\varepsilon^{\mathsf{enc}}_{h,r}$, $\varepsilon^{2}_{h,r}$	Encoding and squared-norm failure allocations when separate marginal bounds are used
$\rho_{\mathsf{model},h}$	Multiplicative envelope relating the planner's modeled response moment to its stated completeness premise
M_h	Typed recursive moment state $(d_{\mathsf{pack}}; (\mu_\chi, v^{\mathsf{full}}_\chi, v^{\mathsf{local}}_\chi)_{\chi \in \mathsf{RespPart}})$
$\alpha \in E$	Ring-reduction evaluation challenge, shared by quotient lifting and quotient-free checking
τ_1	Row-batching challenge point
τ_0	Range point: uniform point of the range identity
$\vartheta_{g,c}$	Public weight that combines claim c of source group g in a multi-instance fold
ζ_{batch}	Fresh challenge whose powers derive the public root-batching weights
b, b_1, b_z	Positional gadget bases for source, opening-partial, and folded-response recomposition
$b^\star$	Common certified range base whose balanced alphabet contains every digit alphabet in one recursive witness
$N_{\mathsf{clm},h}$, $N_{\mathsf{chunk},h}$	Total claims entering receiver h and response chunks retained by fold h
$\Delta^{\mathsf{cert}}_{f,g}$, $\Delta^{\mathsf{cert}}_{\mathsf{rsp},h,g,a}$, $\Delta^{\mathsf{cert}}_{\Sigma,h,g}$	Certified difference envelopes for one group response, one retained response chunk, and their chunk aggregate
δ_{com}, δ_{open}, δ_f, δ_r	Tight digit depths for commitment, opening, fold response, and quotient

B Multilinear Evaluation from Structured Factors

Each sum-check of a fold ends with public weight polynomials that the verifier evaluates at a random point. Computing such a weight from its values on the Boolean cube costs as much as the witness or the setup, which is the work a succinct verifier must avoid. This appendix gives a general method that evaluates these weights from their structure. Appendix C applies it to Akita's opening, range, norm, and setup weights, and also derives the residue kernels of quotient-free ring checking.

The weights are products of simple factors in their logical coordinates: a block index selects a folding challenge, a digit index selects a gadget power, and an opening coordinate selects an equality weight. The difficulty is where these entries sit in the committed vector. Offsets, strides that are not powers of two, and interleaved coordinates need not respect the bit boundaries of the factors, so the usual factorization of the equality polynomial does not apply directly.

We read the address from its low bits to its high bits and keep only what the bits already read determine about the bits still to come. For an offset this is an addition carry; a truncated interval adds a comparison bit. Appendix B.1 works an example, Appendix B.2 states the general problem, Appendix B.3 solves it by a dynamic program over these states, and Appendices B.4 and B.5 specialize it to the affine addresses that Akita uses.

B.1 From Tensor Factors to Vector Addresses

Split the logical index into blocks $x = (x_1, \dots, x_k)$ with $x_t \in \mathcal{X}_t$, and let its weight be $u(x) = \prod_{t=1}^{k} u_t(x_t)$. An address map A places each logical entry at an n-bit vector address; throughout this appendix, an integer address is identified with its little-endian bit vector. We want

$$T_A(\mathbf{r}; u) := \sum_{x \in \mathcal{X}_1 \times \cdots \times \mathcal{X}_k} \left(\prod_{t=1}^{k} u_t(x_t) \right) \widetilde{\mathsf{eq}}(\mathbf{r}, A(x)). \tag{250}$$

This is a multilinear evaluation: if $U_A(y) := \sum_{x : A(x) = y} u(x)$ is the vector that places each weight at its address, then $T_A(\mathbf{r}; u) = \widetilde{U}_A(\mathbf{r})$. Entries with the same address add their weights. Enumerating all logical entries costs $\prod_t |\mathcal{X}_t|$ operations; we want to work with the factors u_t separately whenever the address map allows it.

Aligned coordinates. Suppose $A(x)$ concatenates blocks $A_t(x_t)$ on disjoint sets J_t of address bits. The equality weight then factors, and

$$T_A(\mathbf{r}; u) = \prod_{t=1}^{k} \left(\sum_{x_t \in \mathcal{X}_t} u_t(x_t) \widetilde{\mathsf{eq}}(\mathbf{r}_{J_t}, A_t(x_t)) \right).$$

Once the local equality weights are available, this costs $O(\sum_t |\mathcal{X}_t|)$ operations.

An offset links the factors through a carry. Write $x = x_{\mathsf{lo}} + 4x_{\mathsf{hi}}$ with $x_{\mathsf{lo}} \in [4]$, let $u(x) = u_{\mathsf{lo}}(x_{\mathsf{lo}}) u_{\mathsf{hi}}(x_{\mathsf{hi}})$, and take $A(x) = 3 + x$, assuming all these addresses fit the vector domain. The low two address bits are those of $(3 + x_{\mathsf{lo}}) \bmod 4$, and the high bits are those of $x_{\mathsf{hi}} + c$, where $c = \lfloor (3 + x_{\mathsf{lo}})/4 \rfloor \in \{0, 1\}$ is the carry. Summing the low block separately for each carry gives two scalars

$$\Lambda_c := \sum_{\substack{x_{\mathsf{lo}} \in [4] \\ \lfloor (3 + x_{\mathsf{lo}})/4 \rfloor = c}} u_{\mathsf{lo}}(x_{\mathsf{lo}}) \widetilde{\mathsf{eq}}\big(\mathbf{r}_{\mathsf{lo}}, (3 + x_{\mathsf{lo}}) \bmod 4\big),$$

and the high block needs only the carry:

$$T_A(\mathbf{r}; u) = \sum_{c=0}^{1} \Lambda_c \sum_{x_{\mathsf{hi}}} u_{\mathsf{hi}}(x_{\mathsf{hi}}) \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{hi}}, x_{\mathsf{hi}} + c).$$

The offset replaces one product by a sum of two products. Only the carry passes from the low block to the high block, not the low index itself. Appendix B.3 turns this observation into a general algorithm.

B.2 The General Problem

Some verifier weights involve two addresses at once. An opening weight, for example, is indexed by a logical position, while its contribution to the final sum-check is indexed by a cell of the committed witness, and both give equality factors. We therefore allow several address maps, each with a product of bit weights. For $a \in [p]$, let $A_a(x)$ be an n_a-bit address and

$$B_a(y) := \prod_{j=0}^{n_a - 1} b_{a,j}(y_j), \qquad b_{a,j} : \{0, 1\} \to \mathbb{F}.$$

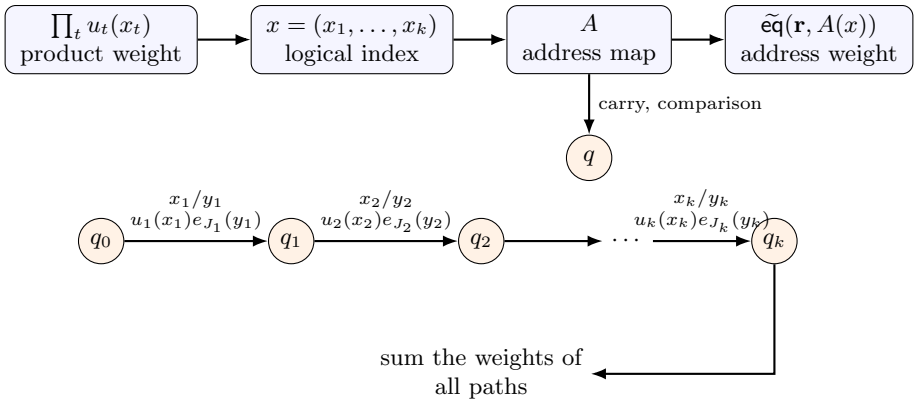


Fig. 13: The logical factors need not align with the address bits of the equality polynomial. The state q records how the blocks read so far affect the address bits still to come, and $e_J(y) = \widetilde{\mathsf{eq}}(\mathbf{r}_J, y)$. Block t is read as input x_t and writes address bits y_t; the bottom row is both a dynamic program and a layered weighted branching program.

Equality at $\mathbf{r}_a$ is the choice $b_{a,j}(0) = 1 - r_{a,j}$, $b_{a,j}(1) = r_{a,j}$. The choice $b_{a,j}(0) = 1$, $b_{a,j}(1) = z_{a,j}$ gives monomials, and $b_{a,j}(0) = 1$, $b_{a,j}(1) = \alpha^{2^j}$ gives the geometric weight α^y. The address calculation is the same for all of them.

Let $D \subseteq \mathcal{X}_1 \times \cdots \times \mathcal{X}_k$ be the logical domain of the sum, which records the included intervals and omitted coordinates. The general problem is to compute

$$T := \sum_{x \in D} \left(\prod_{t=1}^{k} u_t(x_t) \right) \prod_{a \in [p]} B_a(A_a(x)). \tag{251}$$

Every address $A_a(x)$ with $x \in D$ lies in $[2^{n_a}]$. An address outside its domain gets weight zero; addresses never wrap modulo 2^{n_a}.

The applications often use two equality weights:

$$\sum_{x \in D} u(x) \widetilde{\mathsf{eq}}(\mathbf{r}_L, A_L(x)) \widetilde{\mathsf{eq}}(\mathbf{r}_R, A_R(x)). \tag{252}$$

Concatenating A_L and A_R makes this a single-address instance, but two maps make the two carries easier to follow. Sums of several instances of (251), with different domains, weights, and offsets, represent a union of witness intervals or several matrix views of the same setup coefficient.

The cost also depends on how each factor u_t is given. A dense factor is given by its list of values, which must be read. A factor on a Boolean block may instead be given as a product over its bits, $u_t(x_t) = \prod_j w_{t,j}(x_{t,j})$; this description is linear in the number of bits while the list of values is exponential, and the evaluation uses the description directly.

B.3 Finite-State Contraction

We now formalize the state passed between logical blocks. The output bits may belong to any of the address maps in (251); concatenate all their coordinates into one index set J, write b_j for the bit weight of coordinate $j \in J$, and partition J into sets $J_1, \ldots, J_k$. A set J_t may be empty when block t outputs no address bits yet. A last block with a one-element input set and weight one outputs the remaining carry bits without reading a logical coordinate.

Definition B.1 (Weighted address transducer). *A weighted address transducer consists of a finite state set Q, an initial state q_{in}, final weights $\omega_{\mathsf{fin}} : Q \to \mathbb{F}$, and transition sets*

$$\Gamma_t \subseteq Q \times \mathcal{X}_t \times Q \times \{0,1\}^{J_t} \times \mathbb{F}, \qquad t \in [1, k].$$

A transition $(q, x_t, q', y_t, \eta_t)$ reads x_t, outputs y_t, moves from q to q', and has scalar weight η_t. A path for x is a sequence of transitions $(q_{t-1}, x_t, q_t, y_t, \eta_t) \in \Gamma_t$ for $t = 1, \dots, k$ with $q_0 = q_{\mathsf{in}}$. The transducer assigns to x the total weight of its paths:

$$v(x) := \sum_{\text{paths for } x} \omega_{\mathsf{fin}}(q_k) \prod_{t=1}^{k} \left(\eta_t \prod_{j \in J_t} b_j(y_{t,j}) \right). \tag{253}$$

It is deterministic if each pair (q, x_t) has at most one outgoing transition, written $\tau_t(q, x_t) = (q', y_t, \eta_t)$.

For a deterministic address calculation, set every transition scalar to one, let the transitions output the address bits, and accept exactly the inputs in D; an undefined transition or a zero final weight rejects. Then $v(x) = \mathbf{1}_D(x) \prod_a B_a(A_a(x))$, and $T = \langle u, v \rangle$. Nondeterministic transducers allow several logical contributions to share an address.

The dynamic program keeps one field element per state. Set $z_0(q_{\mathsf{in}}) = 1$ and $z_0(q) = 0$ otherwise. At block t, start from the zero vector z_t and, for each transition in Γ_t, add

$$z_t(q') \mathrel{+}= z_{t-1}(q) u_t(x_t) \eta_t \prod_{j \in J_t} b_j(y_{t,j}). \tag{254}$$

Paths that reach the same state have the same continuations, so their weights can be added before the next block is read.

Theorem B.2 (Finite-state structured inner product). *The recurrence (254) gives*

$$\sum_{q \in Q} z_k(q) \omega_{\mathsf{fin}}(q) = \langle u_1 \otimes \cdots \otimes u_k, v \rangle .$$

Let $G_t = |\Gamma_t|$ and $G_{\mathsf{fin}} = |\operatorname{supp}(\omega_{\mathsf{fin}})|$. Given the local factor values and bit weights, the recurrence uses $O(\sum_t G_t(|J_t| + 1) + G_{\mathsf{fin}})$ field operations and stores $O(|Q|)$ field elements for the current and next state vectors. Exactly, multiplying along each transition with no shortcut for zero or unit factors uses

$$\sum_t G_t(|J_t| + 2) + G_{\mathsf{fin}} \quad \textit{multiplications and} \quad \sum_t G_t + \max(G_{\mathsf{fin}} - 1, 0) \quad \textit{additions.}$$

These counts exclude initialization, construction of the transitions, and preparation of the local weights. If the output-bit products are prepared in time P_{eq} and stored for lookup, the bound becomes $O(P_{\mathsf{eq}} + \sum_t G_t + G_{\mathsf{fin}})$ field operations, and their storage comes in addition to the state vectors.

Proof. By induction on t, $z_t(q)$ is the total weight of the path prefixes through block t that end at q, where the weight of a prefix is the product of its logical factors, transition scalars, and output-bit weights. This holds for $t = 0$ and is preserved by (254). Multiplying by the final weights and summing over states gives (253) summed against $u(x)$. Each transition multiplies by its logical factor, its scalar, and its $|J_t|$ bit weights, and adds the result to the next state vector; the final dot product gives the remaining operations.

The theorem bounds the cost by the size of the transducer; it does not claim that every address map has a small transducer. For a deterministic map, $G_t \le |Q|\,|\mathcal{X}_t|$, so a small state set gives cost linear in the sum of the factor sizes. Several carries need their joint state, and a dense factor cannot be replaced by bit weights; the affine cases below count these costs explicitly. Prepared equality weights can be reused for further evaluations at the same point; a new point needs new weights.

Relation to prior evaluation algorithms. The recurrence of Theorem B.2 is the standard evaluation of a weighted read-once branching program: each logical symbol selects a sparse state transition, and distributivity sums out its factor before the transitions are composed. Holmgren and Rothblum showed that this computes the low-degree extension of a size-S read-once branching program in $O(S)$ ring operations [53]; the theorem above allows arbitrary local factor values rather than only the Lagrange weights of one point.

Jagged polynomial commitments use the same calculation for a width-4 program that tracks one addition carry and one comparison bit [52], the closest cryptographic precedent for Akita's carry states. The transfer-matrix view is the classical equivalence between weighted automata and matrix-product representations [36]. What is specific to Akita is that its unaligned witness and setup layouts have small-state address transducers, and that the weights can be evaluated with the exact transition counts above.

B.4 Affine Addresses on Intervals

For affine layouts the state is an integer carry. We start with consecutive digits, then allow separate strides and several address maps, so that the cost of the digits and the cost of the layout appear separately.

Consecutive digits. When each logical position i occupies δ consecutive digit cells starting at address o, digit ℓ of position i has address

$$A(i,\ell) = o + \delta i + \ell, \qquad i \in [L], \quad \ell \in [\delta]. \tag{255}$$

Neither L nor δ needs to be a power of two. We assume all these addresses lie in $[2^n]$.

Lemma B.3 (Reading consecutive-digit addresses). *The bits of (255) can be read from low to high with a carry taking $O(\delta)$ values at each layer. The calculation can also enforce $i < L$, and it never divides by a coordinate of the evaluation point.*

Proof. Read ℓ first and set the carry to $c_0 = o + \ell$. At bit t of i, write $c_t + \delta i_t = y_t + 2c_{t+1}$ with $y_t \in \{0,1\}$. By induction,

$$c_t = \left\lfloor \frac{o + \ell + \delta(i \bmod 2^t)}{2^t} \right\rfloor,$$

so the output bits are those of $o+\ell+\delta i$. As ℓ and the bits read so far vary, c_t ranges over an interval of $O(\delta)$ integers. After the bits of i end, output the bits of the carry. To enforce $i < L$, pad i and L to a common width and also keep the comparison of the bits of i read so far with the corresponding bits of L: a new, more significant unequal bit decides the comparison, and otherwise the previous comparison stands. This multiplies the state by three.

Summing a low block before reading the high block. The bit recurrence needs factors that split over bits, which an arbitrary factor on a block does not. Instead, we can sum a whole low block into one bucket per carry, as in the example of Appendix B.1, and then scan the high factor once per bucket.

Proposition B.4 (Consecutive-digit evaluation). *Let $L, \delta \geq 1$, $o + \delta L \leq 2^n$, and $m \leq n$. Write $i = 2^m i_{\mathsf{hi}} + i_{\mathsf{lo}}$ with $i_{\mathsf{lo}} \in [2^m]$, and suppose $\psi(i) = \psi_{\mathsf{hi}}(i_{\mathsf{hi}})\psi_{\mathsf{lo}}(i_{\mathsf{lo}})$ for $i < L$. Let $\phi : [\delta] \to \mathbb{F}$ be the digit weights. Given the local equality weights for constant-time lookup, the sum*

$$T = \sum_{i<L} \sum_{\ell<\delta} \psi(i)\phi(\ell)\widetilde{\mathsf{eq}}(\mathbf{r}, o + \delta i + \ell)$$

can be computed in $O(\delta(2^m + \lceil L/2^m \rceil))$ field operations using $O(\delta)$ accumulators, in addition to the stored factor values and equality weights. Preparing the equality weights in time P_{eq} gives total work $O(P_{\mathsf{eq}} + \delta(2^m + \lceil L/2^m \rceil))$.

Proof. Write $o = 2^m o_{\mathsf{hi}} + o_{\mathsf{lo}}$ with $o_{\mathsf{lo}} \in [2^m]$, and split $\mathbf{r}$ into its m low coordinates $\mathbf{r}_{\mathsf{lo}}$ and the remaining coordinates $\mathbf{r}_{\mathsf{hi}}$. The address of (i,ℓ) is $2^m(o_{\mathsf{hi}} + \delta i_{\mathsf{hi}}) + (o_{\mathsf{lo}} + \delta i_{\mathsf{lo}} + \ell)$. For a complete row of 2^m positions, define the buckets

$$\Lambda_c := \sum_{\substack{i_{\mathsf{lo}}<2^m,\ \ell<\delta \\ \lfloor (o_{\mathsf{lo}}+\delta i_{\mathsf{lo}}+\ell)/2^m \rfloor = c}} \psi_{\mathsf{lo}}(i_{\mathsf{lo}})\phi(\ell)\widetilde{\mathsf{eq}}\big(\mathbf{r}_{\mathsf{lo}}, (o_{\mathsf{lo}} + \delta i_{\mathsf{lo}} + \ell) \bmod 2^m\big).$$

Since $o_{\mathsf{lo}} + \delta i_{\mathsf{lo}} + \ell < (\delta+1)2^m$, the carry satisfies $0 \leq c \leq \delta$, so there are at most $\delta+1$ buckets. The complete rows contribute

$$\sum_{i_{\mathsf{hi}} < \lfloor L/2^m \rfloor} \psi_{\mathsf{hi}}(i_{\mathsf{hi}}) \sum_{c=0}^{\delta} \Lambda_c\, \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{hi}}, o_{\mathsf{hi}} + \delta i_{\mathsf{hi}} + c).$$

The buckets take $O(\delta 2^m)$ operations and the high sum $O(\delta \lfloor L/2^m \rfloor)$. If the last row is partial, form a second set of buckets restricted to $i_{\mathsf{lo}} < L \bmod 2^m$ and combine them with that row's high weight, for another $O(\delta 2^m)$ operations.

The factor δ in the bound is the number of carry values, not a hidden constant. Preparing the equality weights is not free either: evaluating an equality weight directly costs one operation per bit, up to constant factors, and Appendix B.5 shares this work. An arbitrary vector $(\psi(i))_{i<L}$ has L independent entries and needs $\Omega(L)$ reads; the saving comes from the given split $\psi = \psi_{\mathsf{hi}}\psi_{\mathsf{lo}}$.

Separate strides and partial windows. The same calculation applies to

$$A(i,\ell) = o + s(i - i_0) + s_{\mathsf{dig}}\ell, \qquad i_0 \le i < i_0 + L, \quad \ell \in [\delta], \tag{256}$$

with positive integer strides satisfying $s > s_{\mathsf{dig}}(\delta - 1)$, so that the digits of one position end before the next position begins. Here o is the address of the first position i_0, while the outer weight $\psi_{\mathsf{hi}}(i_{\mathsf{hi}})\psi_{\mathsf{lo}}(i_{\mathsf{lo}})$ uses the global index $i = 2^m i_{\mathsf{hi}} + i_{\mathsf{lo}}$. The two differ when a witness interval begins partway through a logical row.

Split the window into a partial first row, complete rows, and a partial last row, counting a row once if the whole window lies in it. If the included part of row i_{hi} begins at low index $i_\star$, its first address is $o' := o + s(2^m i_{\mathsf{hi}} + i_\star - i_0)$. Form the buckets of that part from the low addresses $(o' \bmod 2^m) + s(i_{\mathsf{lo}} - i_\star) + s_{\mathsf{dig}}\ell$ and the weights $\psi_{\mathsf{lo}}(i_{\mathsf{lo}})\phi(\ell)$; there are at most $s + 1$ of them. For complete rows, $i_\star = 0$ and successive first addresses differ by $s2^m$, so all complete rows share one set of buckets and differ only in the high address. If R high rows meet the window, the evaluation costs

$$O\big(\delta 2^m + (s+1)R\big)$$

with equality lookup, plus preparation. An empty window contributes zero. A large outer stride thus enlarges the state even when there are few digits.

Remark B.5 (Power-of-two special case). If the digit count is a power of two and the starting address is a multiple of it, the equality polynomial factors across the digit boundary. An unaligned start adds a carry across that boundary. Both cases evaluate the same stored vector; no padding or moving of entries is needed.

Several address maps. Let $x_1, \dots, x_k$ be nonnegative integer coordinates and

$$A_a(x) = o_a + \sum_{t=1}^{k} s_{a,t} x_t \qquad (a \in [p]),$$

with nonnegative strides; a zero stride lets an axis affect only some of the addresses. When each x_t ranges over a Boolean block and its weight factors over bits, read bit j of every axis together. Set $c_{a,0} = o_a$ and update

$$c_{a,j} + \sum_{t} s_{a,t} x_{t,j} = y_{a,j} + 2c_{a,j+1}, \qquad y_{a,j} \in \{0,1\}. \tag{257}$$

Multiply the state weight by the input-bit weights and by $\prod_{a:\, j<n_a} b_{a,j}(y_{a,j})$, then add the weights of equal carry tuples. After an axis ends, its remaining bits are zero. After an address ends, its remaining output bits and its final carry must be zero; this enforces the address domains instead of wrapping.

As in Lemma B.3, the carry of address a ranges over at most $S_a + 1$ integers at each layer, where $S_a = \sum_t s_{a,t}$. The joint state therefore has at most $\prod_a (S_a + 1)$ carry tuples before comparison states are added, and often far fewer are reachable. If k_j axes have a bit at layer j, each state has at most 2^{k_j} transitions there. Theorem B.2 counts these transitions and their local weights, so the cost depends on both the strides and the number of axes read together.

An axis of arbitrary length can be split into aligned power-of-two intervals, each with its own base address, or its length bound can be kept as a comparison state. A dense axis with no useful bit factorization can be enumerated, each value starting an affine evaluation of the remaining axes. Enumerating several such axes costs the product of their lengths, so we do this only for axes short enough for the verifier bound.

B.5 Sharing Equality Work and Combining Families

The bounds above count the dynamic program and the preparation of its local weights separately. We now give simple ways to reduce the preparation and to share one evaluation among several contributions.

Offset equality on an interval. For a stored vector $f \in \mathbb{F}^L$, a scale $\theta \in \mathbb{F}$, and an offset o, consider

$$\theta \sum_{z=0}^{L-1} f[z]\widetilde{\mathsf{eq}}(\mathbf{r}, o+z). \tag{258}$$

Addresses outside $[2^n]$ contribute zero. If no address lies in $[2^n]$, the sum is zero; otherwise let $[\mathsf{lo}, \mathsf{hi}]$ be the inclusive endpoints of the addresses that do. Fold the vector one coordinate at a time, starting from the low bit, and merge only parents with at least one child whose subtree meets $[\mathsf{lo}, \mathsf{hi}]$. The number of merges is

$$\sum_{j=0}^{n-1} \left(\left\lfloor \frac{\mathsf{hi}}{2^{j+1}} \right\rfloor - \left\lfloor \frac{\mathsf{lo}}{2^{j+1}} \right\rfloor + 1 \right).$$

A merge of children v_0, v_1 computes $v_0 + r_j(v_1 - v_0)$, with a missing child read as zero, so each merge costs one multiplication, and one more multiplication applies θ. This evaluates arbitrary stored weights on an interval without extending them to the whole vector domain.

Prepared equality factors. When many addresses share one evaluation point, split its coordinates into a low and a high block. Tabulating the equality weights of each block takes $O(2^{n_{\mathsf{lo}}} + 2^{n_{\mathsf{hi}}})$ field operations and storage, after which each equality weight is a product of two lookups. A balanced split stores square-root many weights. With less storage, tabulate only a short block and evaluate the other bits on demand, at a cost linear in their number per lookup. The same tradeoff applies to products of arbitrary bit weights.

One carry from a power-of-two low block. For $x = x_{\mathsf{lo}} + 2^m x_{\mathsf{hi}}$, an offset carries at most one from the low block. An arbitrary low factor can be summed into two buckets in $O(2^m)$ operations, given its equality weights. If the low factor is itself an equality weight, read its bits instead: multiply each bit's logical equality weight by the equality weight of the corresponding output bit and keep the addition carry. This two-state recurrence takes $O(m)$ field operations and never lists either weight vector.

Geometric digit weights. Gadget powers give a further saving. Let the digit weights be $\phi(\ell) = \phi(0)\gamma^\ell$ with $\gamma \neq 0$, and let $(e_{\mathsf{lo}}(v))_{v<2^m}$ be the tabulated low-block equality weights. With the prefix sums

$$\mathrm{Pre}(v) := \sum_{z<v} \gamma^z e_{\mathsf{lo}}(z),$$

a digit window that starts at v_0 and ends before 2^m satisfies

$$\sum_{\ell<\delta} \gamma^\ell e_{\mathsf{lo}}(v_0+\ell) = \gamma^{-v_0}\big(\mathrm{Pre}(v_0+\delta) - \mathrm{Pre}(v_0)\big).$$

With the needed powers and inverse powers of γ stored next to Pre, each window costs constant work, and multiplying by $\phi(0)$ restores the digit weights. If $\delta \le 2^m$, a window that crosses the low-block boundary splits into two pieces, each added to its carry bucket. For unit digit stride and a fixed residue of the base address, this reduces the bucket computation from $O(\delta 2^m)$ to $O(\delta + 2^m)$, including the prefix sums. A zero ratio or a non-geometric sequence uses the direct computation. No coordinate of the evaluation point is inverted.

Sharing across offsets and summed families. Consecutive-digit families with the same strides, factors, and residue of the base address modulo 2^m have the same buckets on the same range of low indices. Compute those buckets once; the families' different high offsets and scalar weights then initialize one shared high-state vector. Adding the weights of equal states before continuing is valid at every layer because the recurrence is linear. Likewise, paired evaluations whose remaining transitions coincide can share their carry states. The cost is that of preparing the initial contributions, including forming and grouping them, plus the transitions actually processed.

Aligned families are simpler still. If distinct axes occupy disjoint bit ranges in every address and the offsets produce no carries into those ranges, factor them as in Appendix B.1. If the address maps also keep the bits within an axis separate, a constant or bit-product factor on that axis costs work linear in its bit length, while a dense axis requires reading its entries and their local weights. Without alignment, keep the carry recurrence. Every method computes the same value $T_A(\mathbf{r}; u) = \widetilde{U}_A(\mathbf{r})$.

C Evaluating Akita's Public Weights

This appendix computes the public weights that Akita's verifier evaluates at the end of a fold. Wherever a weight factors over logical coordinates but sits at shifted or interleaved witness addresses, we use the evaluator of Appendix B. Appendix C.1 first derives the weights of quotient-free ring checking, which involve reduction modulo the ring polynomial and so are not plain powers of α. Appendices C.2 and C.3 then give the factors of the opening, range, norm, and relation weights; Appendix C.4 combines the overlapping setup views, both for a direct scan and after setup offloading; and Appendix C.5 totals the verifier cost.

C.1 Quotient-Free Relation Weights

Fix a native ring dimension d, a public multiplier $A(X) = \sum_{s<d} a_s X^s$, and a point $\alpha \in E$. The residue kernel (117) is the projected rotation-matrix row of the concurrent Grand Danois construction [57]. To state this precisely, define the signed rotation

$$\mathrm{rot}(v_0, \ldots, v_{d-1}) := (-v_{d-1}, v_0, \ldots, v_{d-2}), \tag{259}$$

and let $\mathsf{cf}_d(A) := (a_0, \ldots, a_{d-1})^\top$. Column j of

$$\mathsf{M}_A := \left[\mathsf{cf}_d(A) \mid \mathrm{rot}(\mathsf{cf}_d(A)) \mid \cdots \mid \mathrm{rot}^{d-1}(\mathsf{cf}_d(A))\right] \tag{260}$$

is the coefficient vector of $AX^j \bmod (X^d + 1)$. Hence, with $\boldsymbol{p}_d(\alpha)^\top := (1, \alpha, \ldots, \alpha^{d-1})$,

$$\left(\boldsymbol{p}_d(\alpha)^\top \mathsf{M}_A\right)_j = \kappa^{(d)}_{A,\alpha}(j). \tag{261}$$

The two are the same linear map for one ring product. The residue-kernel form fits the common row normal form, in which row families may have different native dimensions and noncontiguous witness address maps.

Computing each of the d kernel values from (118) would cost $O(d^2)$ operations. Consecutive basis products differ by a factor of X, which gives a linear-time recurrence.

Lemma C.1 (Kernel recurrence). *The residue kernel of $A(X) = \sum_{s<d} a_s X^s$ satisfies*

$$\kappa^{(d)}_{A,\alpha}(0) = A(\alpha), \tag{262}$$

$$\kappa^{(d)}_{A,\alpha}(j+1) = \alpha \kappa^{(d)}_{A,\alpha}(j) - (\alpha^d + 1) a_{d-1-j}, \qquad j \in [d-1]. \tag{263}$$

Consequently, one kernel takes $O(d)$ extension-field operations and no division by $\alpha^d + 1$.

Proof. The initial value holds because A has degree less than d. Multiplying $AX^j \bmod (X^d + 1)$ by X shifts every coefficient up by one place, and a_{d-1-j} is the only coefficient that reaches degree d. Evaluating the shifted polynomial multiplies by α, and replacing α^d by -1 in that one term subtracts $(\alpha^d + 1)a_{d-1-j}$, which gives (263).

Where the quotient went. Euclidean division of each public basis product gives a public *basis quotient* $q^{\mathsf{basis}}_{A,j}$ with

$$A(X)X^j = \left(A(X)X^j \bmod (X^d + 1)\right) + (X^d + 1) q^{\mathsf{basis}}_{A,j}(X). \tag{264}$$

Evaluating at α shows that the kernel already subtracts the quotient term:

$$\kappa^{(d)}_{A,\alpha}(j) = A(\alpha)\alpha^j - (\alpha^d + 1) q^{\mathsf{basis}}_{A,j}(\alpha). \tag{265}$$

For a whole row, the quotient of (111) is therefore

$$Q_r(X) = \sum_{c \in \mathsf{Occ}_r} \sum_{j=0}^{d_r - 1} w_{\mathsf{addr}_{r,c}(j)} q^{\mathsf{basis}}_{A_{r,c},j}(X), \tag{266}$$

a linear combination of witness coefficients with public weights. The kernel includes these weights, so the check accounts for the reduction without a quotient from the prover.

The Quotient-Free Relation at the Sum-Check Point

Batched field relation. After reduction, each ring row is a linear equation in the witness coefficients, and the scalar opening rows are already of this form. Let $\mathsf{Rows}_{\mathsf{field}}$ be the ordered set of scalar rows, written

$$z_r^{\mathsf{fld}} := \sum_{u\in[N_{\mathsf{w}}]} b_{r,u} w_u - y_r, \tag{267}$$

where the public coefficients $b_{r,u} \in E$ are the direct or evaluation-trace opening weights. With n_{rel} ring and scalar rows in total and $s_{\mathsf{rel}} := \lceil \log_2 n_{\mathsf{rel}} \rceil$, pad the rows with zero rows to $2^{s_{\mathsf{rel}}}$. The row-batching point $\tau_1 \in E^{s_{\mathsf{rel}}}$ gives row r the weight

$$\omega_{\mathsf{row},r}(\tau_1) := \widetilde{\mathsf{eq}}(\tau_1, r). \tag{268}$$

The coefficient of witness entry u in the batched relation sums the contributions of every row that reads it:

$$\begin{aligned} m^{\mathsf{qf}}_{\alpha,\tau_1}(u) := &\sum_{r\in\mathsf{Rows}_{\mathsf{ring}}} \omega_{\mathsf{row},r}(\tau_1) \sum_{\substack{c\in\mathsf{Occ}_r,\ j\in[d_r]\\ \mathsf{addr}_{r,c}(j)=u}} \kappa^{(d_r)}_{A_{r,c},\alpha}(j) \\ &+ \sum_{r\in\mathsf{Rows}_{\mathsf{field}}} \omega_{\mathsf{row},r}(\tau_1) b_{r,u}, \qquad u \in [2^{\mu'}], \end{aligned} \tag{269}$$

where every summand with $u \geq N_{\mathsf{w}}$ is zero. The public target is

$$V^{\mathsf{qf}}_{\alpha,\tau_1} := \sum_{r\in\mathsf{Rows}_{\mathsf{ring}}} \omega_{\mathsf{row},r}(\tau_1) Y_r(\alpha) + \sum_{r\in\mathsf{Rows}_{\mathsf{field}}} \omega_{\mathsf{row},r}(\tau_1) y_r. \tag{270}$$

These definitions give (121). Because (269) sums every use of a witness coordinate, it remains correct when row families with different native dimensions read overlapping coordinates.

Evaluation at the sum-check point. The prover can list the coefficient vector (269); the verifier needs only its multilinear extension at the final sum-check point $\mathbf{r}_x$. For one occurrence (r, c), define the equality weights of its witness window

$$e_{r,c,\mathbf{r}_x}(j) := \widetilde{\mathsf{eq}}\big(\mathbf{r}_x, \mathsf{addr}_{r,c}(j)\big), \qquad j \in [d_r], \tag{271}$$

where, as elsewhere, an integer second argument of $\widetilde{\mathsf{eq}}$ stands for its μ'-bit encoding. The definition uses the witness address map, so the window need not start at zero or occupy a whole tensor coordinate.

Exchanging the two sums in (118) moves these equality weights onto the coefficients of A. For a sequence $e = (e_0, \ldots, e_{d-1}) \in E^d$, define the *transposed residue kernel*

$$\kappa^{(d),\top}_{e,\alpha}(s) := \sum_{j=0}^{d-1} e_j (-1)^{\lfloor (s+j)/d \rfloor} \alpha^{(s+j) \bmod d}. \tag{272}$$

Then the contribution of the occurrence is

$$\sum_{j=0}^{d-1} e_j \kappa^{(d)}_{A,\alpha}(j) = \sum_{s=0}^{d-1} a_s \kappa^{(d),\top}_{e,\alpha}(s), \tag{273}$$

and the transposed kernel satisfies the same recurrence as Lemma C.1, with the roles of the two sequences exchanged:

$$\kappa^{(d),\top}_{e,\alpha}(0) = \sum_{j=0}^{d-1} e_j \alpha^j, \tag{274}$$

$$\kappa^{(d),\top}_{e,\alpha}(s+1) = \alpha \kappa^{(d),\top}_{e,\alpha}(s) - (\alpha^d + 1) e_{d-1-s}, \qquad s \in [d-1]. \tag{275}$$

The verifier computes one transposed kernel for each distinct pair of native dimension and witness window, and takes its inner product with every public multiplier on that window. This replaces the power tensor $\widetilde{\alpha}$ of (25) on the quotient-free route.

C.2 Opening, Range, and Norm Weights

For each of these checks we give the logical weights, the witness address, and the domain of the sum; the evaluator of Appendix B then computes the weight without listing its values on the witness.

Opening rows. With direct coefficient packing, the factors are those of (94), placed at the addresses of the opening digits. Let o_e be the first coefficient of the opening-digit interval and $s_e := kd_{\mathsf{car}}/d_D$, and split the packed coefficient index as $j + d_{\mathsf{car}}t = d_D v + u$ with $u \in [d_D]$. The native-block order of Section 5.5 and Appendix L.2 places digit ℓ of cell (i,t,j) at

$$\mathsf{addr}_e(i,\ell,t,j) := o_e + \big((is_e + v)\delta_{\mathsf{open}} + \ell\big)d_D + u. \tag{276}$$

At the final point $\mathbf{r}_x$ of the relation sum-check, the verifier needs

$$\widetilde{\omega}_{\mathsf{dir}}(\mathbf{r}_x) = \sum_{i,\ell,t,j} \omega_{\mathsf{dir}}(i,\ell,t,j)\,\widetilde{\mathsf{eq}}\big(\mathbf{r}_x, \mathsf{addr}_e(i,\ell,t,j)\big). \tag{277}$$

If the opening-digit interval is aligned and the digit depth is a power of two, the equality weight factors along the block, subcolumn, digit, and coefficient axes. Otherwise Proposition B.4 handles the carries of (276). Either way, the verifier never forms the $B\delta_{\mathsf{open}}kd_{\mathsf{car}}$ entries of the row.

Evaluation-trace weights. Let $\varphi_\nu := \mathcal{T}_{\rho_{\mathsf{pack}}}(X^\nu)$ for $\nu \in [d_A]$, with the functional of Section 7.1. By (99), the scaled weight on opening digit (i,ℓ,ν) is

$$\theta\,\omega_{\mathsf{Tr}}(i,\ell,\nu) = \theta\widetilde{\mathsf{eq}}(\rho_{\mathsf{blk}}, i)b^\ell_{\mathsf{op}}\varphi_\nu,$$

where θ is the checked tensor-reduction factor, equal to one for an unscaled claim. The weight sits at the addresses of the opening digits and is zero elsewhere. The values φ_ν depend on the opening point and the packing, but not on the block or the witness.

First suppose the opening interval is aligned, B is a power of two, and the digit axis of each block is padded from δ_{open} to $\Delta = 2^{\lceil \log_2 \delta_{\mathsf{open}} \rceil}$ with zero weights. The witness address bits then consist of fixed bits that select the opening interval, the block index, the subcolumn, the digit index, and the coefficient within a subcolumn. Split $\mathbf{r}_x$ accordingly into $\mathbf{r}_{\mathsf{int}}$, $\mathbf{r}_{\mathsf{blk}}$, $\mathbf{r}_{\mathsf{dig}}$, and $\mathbf{r}_{\mathsf{coef}}$, where $\mathbf{r}_{\mathsf{coef}}$ collects the subcolumn and coefficient coordinates in the order of ν. The weight factors as

$$\begin{aligned}
&\widetilde{\omega}_{\mathsf{Tr}}(\mathbf{r}_x) = \theta\widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{int}}, \mathsf{int})\widetilde{\mathsf{eq}}(\rho_{\mathsf{blk}}, \mathbf{r}_{\mathsf{blk}})D(\mathbf{r}_{\mathsf{dig}})C(\mathbf{r}_{\mathsf{coef}}),\\
&D(\mathbf{s}) := \sum_{\ell<\delta_{\mathsf{open}}} b^\ell_{\mathsf{op}}\widetilde{\mathsf{eq}}(\mathbf{s},\ell), \qquad C(\mathbf{t}) := \sum_{\nu<d_A} \varphi_\nu\widetilde{\mathsf{eq}}(\mathbf{t},\nu),
\end{aligned} \tag{278}$$

where int denotes the fixed interval bits. The block factor uses

$$\begin{aligned}
\sum_{i\in\{0,1\}^{\log B}} \widetilde{\mathsf{eq}}(\rho_{\mathsf{blk}}, i)\widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{blk}}, i) &= \prod_j \big((1-\rho_{\mathsf{blk},j})(1-r_{\mathsf{blk},j}) + \rho_{\mathsf{blk},j}r_{\mathsf{blk},j}\big)\\
&= \widetilde{\mathsf{eq}}(\rho_{\mathsf{blk}}, \mathbf{r}_{\mathsf{blk}}),
\end{aligned}$$

so the verifier never iterates over blocks. Given the values φ_ν, which are computed within one ring, the interval and block factors take $O(\mu')$ field operations, and tabulating the Boolean equality weights gives D and C in $O(\delta_{\mathsf{open}} + d_A)$ operations, including the padding to $\Delta < 2\delta_{\mathsf{open}}$. For this layout the weight therefore costs $O(\mu' + \delta_{\mathsf{open}} + d_A)$ operations, and the work within one ring does not depend on B.

In general, witness intervals may be shifted, block ranges truncated, and digit axes stored without padding, and the product above need not hold. With $\mathsf{addr}_{\mathsf{Tr}}(i,\ell,\nu)$ the witness address of opening digit (i,ℓ,ν), the weight is

$$\widetilde{\omega}_{\mathsf{Tr}}(\mathbf{r}_x) = \theta\sum_{i,\ell,\nu} \widetilde{\mathsf{eq}}(\rho_{\mathsf{blk}}, i)b^\ell_{\mathsf{op}}\varphi_\nu\widetilde{\mathsf{eq}}\big(\mathbf{r}_x, \mathsf{addr}_{\mathsf{Tr}}(i,\ell,\nu)\big).$$

The block weight and the equality weight at the witness address form an instance of (252). Sum the coefficient values within one ring, then apply the affine or paired carry calculation of Appendix B to the block, subcolumn, and digit coordinates. Separate witness intervals contribute additively, each with its own offset and block range.

Restricted equality and response norm weights. The binariness weight of Section 6.1, evaluated at $\mathbf{r}_x$, is

$$\sum_{i \in \mathcal{I}_{\mathsf{bin}}} \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{virt}}, i)\widetilde{\mathsf{eq}}(\mathbf{r}_x, i).$$

Represent each interval $[o, o + L)$ of $\mathcal{I}_{\mathsf{bin}}$ by one logical coordinate $x \in [L]$ with the two maps $A_L(x) = A_R(x) = o + x$. Each interval is a paired affine evaluation with unit logical weights, and the intervals add. This is the multilinear extension of the pointwise product on Boolean addresses; the product of the separate multilinear extensions of the support indicator and the equality weight is a different polynomial.

The norm-linking identities of Section 6.2 have the same form with other maps. One address identifies a coefficient of the reconstructed response or of one of its digit planes, and the other identifies the committed witness cell that holds it. Recomposition powers give the digit factors, the paired recurrence handles the two maps, and the intervals select the participating response cells. The cost therefore depends on the factor descriptions, intervals, and carry states, not on the size of the padded witness domain.

C.3 Factored Blocks in the Relation

Each factored block in Figure 5 uses its native address map from Appendix L.2, including its segment offset. As in Appendix L.1, a block with native dimension d, subcolumn count s, digit depth δ, and offset o places coefficient k of digit u of subcolumn y in row a at

$$\mathsf{addr}(a, y, u, k) = o + \big((as + y)\delta + u\big)d + k.$$

The coefficient k varies fastest, and consecutive digits are d apart. The separate-stride and several-address extensions of Appendix B.4 evaluate these factors with all four coordinates; the consecutive-digit map (255) is the case $d = 1$.

With coefficient packing, the factors are as follows.

– On $\hat{\mathbf{e}}$, the packing-consistency weight of cell (i, ℓ, t, j) is

$$c_i(\alpha)\, \beta_t\, b_{\mathsf{op}}^{\ell}\, \alpha^j,$$

and $c_i(\alpha)$ is shared by all coordinate planes $t \in [k]$.

– On $\hat{\mathbf{z}}$, the carrier row uses the factors of the direct opening contraction: the position weight $\widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{pos}}, x)$, the low-coefficient weight $\widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{pack}}, a)$, the power α^j, and the two gadget digits, at ambient coefficient address $a + kh_{\mathsf{car}}j$.

– On $\hat{\mathbf{t}}$, $\mathbf{A}$ row r uses $c_i(\alpha^{kh_{\mathsf{car}}})$, the $\mathbf{A}$-row selector, and the opening gadget digit. It cannot reuse $c_i(\alpha)$ from the carrier row.

– The direct scalar row on $\hat{\mathbf{e}}$ uses

$$\widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{blk}}, i)\, \beta_t\, \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{tail}}, j)\, b_{\mathsf{op}}^{\ell}$$

and is evaluated by (277).

– With quotient lifting, the carrier quotient cell in plane t at position j has weight $\beta_t\alpha^j$, times $-(\alpha^{d_{\mathsf{car}}} + 1)$ and its quotient gadget digit. Every other quotient family uses its own native power and $-(\alpha^d + 1)$.

– With quotient-free checking there are no quotient cells, and the corresponding cells use the transposed kernels of Section 7.3. Admissible schedules use this route only after the switch to evaluation trace, never at a coefficient-packing fold (Definition 9.2).

With evaluation trace, the factors change as follows. On opening cell (i, ℓ, ν), the fold-evaluation weight is $c_i(\alpha)b_{\mathsf{op}}^{\ell}\alpha^{\nu}$. The response side uses the evaluation-trace opening vector $\mathbf{a}(\alpha)$, and the $\mathbf{A}$ row on $\hat{\mathbf{t}}$ uses the same $c_i(\alpha)$. The scalar row uses $\omega_{\mathsf{Tr}}(i, \ell, \nu)$ from (99); for group g, its unscaled weights $\omega_{\mathsf{Tr},g}$ are multiplied by the public scalar $\vartheta_{g,c}\theta_g$ of (101). Here θ_g contains the group's tensor accumulator at its native prefix and the equality factor on any extra variables of the shared reduction, and equals one for the identity reduction. With quotient lifting, the evaluation-trace fold quotient has the factor $-(\alpha^{d_A} + 1)$.

With quotient-free checking, the same public multipliers enter through the residue kernels (117) instead of their evaluations times a common power of α. At the final sum-check point, the verifier pairs each native occurrence with the equality weights of its window, (271), and applies (273). No separate $\widetilde{\alpha}$ factor appears, since the transposed kernel already sums over the coefficient coordinate.

The compression rows are handled the same way for their gadget and ring-reduction terms, restricted to the scheduled digit intervals. Their matrix terms use the $\mathbf{F}/\mathbf{H}$ matrices, which the verifier evaluates itself, once per distinct shape. This holds at an offloaded level too: these matrices are not part of the $\mathbf{A}/\mathbf{B}/\mathbf{D}$ scan described next.

C.4 Shared Setup Views and Offloaded Weights

The setup blocks are of a different kind: each matrix coefficient comes from the shared setup, not from small public factors. Let $\mathcal{M}_{\mathsf{setup}}$ index the active instances of $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{D}$; the compression matrices $\mathbf{F}/\mathbf{H}$ are handled separately. Instance $I \in \mathcal{M}_{\mathsf{setup}}$ has shape (n_I, m_I, d_I) and reads the first $n_I m_I d_I$ setup coefficients, and the longest of these prefixes has length $N_{\mathsf{active}} := \max_{I \in \mathcal{M}_{\mathsf{setup}}} n_I m_I d_I$. One setup coefficient can belong to several instances. Coefficient p of instance I lies in

$$\mathsf{row}_I(p) := \left\lfloor \frac{p}{m_I d_I} \right\rfloor, \quad \mathsf{col}_I(p) := \left\lfloor \frac{p \bmod (m_I d_I)}{d_I} \right\rfloor, \quad \mathsf{coef}_I(p) := p \bmod d_I,$$

so instances with different ring dimensions give the same coefficient different monomial or residue weights. There is no single setup ring dimension.

The verifier adds the weights of all instances at each p into one weight $\bar{\omega}(p)$, so that $S_p \bar{\omega}(p)$ accounts for every use of S_p and both reduction routes give the scan (126). With quotient lifting,

$$\bar{\omega}^{\mathsf{ql}}(p) := \sum_{\substack{I \in \mathcal{M}_{\mathsf{setup}} \\ p < n_I m_I d_I}} \Omega_I(p)\, \alpha^{\mathsf{coef}_I(p)}, \tag{279}$$

where $\Omega_I(p)$ is the product of the row-batching weight of $\mathsf{row}_I(p)$ and the structured witness weight of $\mathsf{col}_I(p)$.

With quotient-free checking, one coefficient of one instance can contribute through several witness windows. In particular, the same $\mathbf{B}$ matrix serves several logical slices, each with its own witness interval and public weight; $\mathbf{A}$ and $\mathbf{D}$ are unsliced. Each use needs the equality weights of its own window. Let $\mathsf{Occ}_I(p)$ index the pairs of a logical row and a witness window in which coefficient p of instance I takes part. For $\nu \in \mathsf{Occ}_I(p)$, let $e_{I,\nu,\mathbf{r}_x}$ be the window equality weights of (271), and let $\Omega_{I,\nu}(p)$ collect the public row, gadget, slice, and projection factors other than the transposed kernel. Then

$$\bar{\omega}^{\mathsf{qf}}(p) := \sum_{\substack{I \in \mathcal{M}_{\mathsf{setup}} \\ p < n_I m_I d_I}} \sum_{\nu \in \mathsf{Occ}_I(p)} \Omega_{I,\nu}(p) \kappa^{(d_I),\top}_{e_{I,\nu,\mathbf{r}_x},\alpha}\big(\mathsf{coef}_I(p)\big), \tag{280}$$

which is (273) applied to every instance. For $\mathbf{B}$, the inner sum runs over the slices whose unpadded interval contains the column; padding contributes zero. Since overlapping views are combined in $\bar{\omega}$, the verifier scans only the longest prefix.

The combination over slices is already visible in the ring rows. If $\lambda_{s,r}$ weights logical row (s, r), the $\mathbf{B}$ terms combine as

$$\sum_{r \in [n_B]} \sum_{c \in [m_B]} B_{r,c} \left(\sum_{s \in [S_B]} \lambda_{s,r} x_{B,s,c} \right), \tag{281}$$

and at the final sum-check point the public weights combine the same way, with the coefficient weight of the chosen reduction route.

Computing $\bar{\omega}(p)$ is itself work. For $\mathbf{B}$, listing every pair of a slice and a column could be expensive, so we evaluate the slice weights from their structured addresses. For an unsliced matrix, Proposition B.4 gives the column weights in amortized constant field work per column for a fixed digit count, after the local equality weights are prepared. For a sliced matrix, the slice index, its interval, and the consecutive-digit address form one more structured factor, and the evaluator of Appendix B handles it with the same carry state. Computing the sum over slices directly costs $O(S_B)$ field operations per matrix entry; the structured evaluation costs the transition count of Theorem B.2.

The Setup Weight after Offloading After setup offloading, the verifier evaluates $\widetilde{\omega}_{\mathsf{setup}}(\rho_S)$ (Section 9.1). Expanded by its matrix contributions, a typical family has the form

$$\sum_x \left(\prod_t u_t(x_t) \right) \widetilde{\mathsf{eq}}(\rho_S, A_{\mathsf{setup}}(x)) \widetilde{\mathsf{eq}}(\mathbf{r}_x, A_{\mathsf{witness}}(x)),$$

where x ranges over the included row, column, digit, and projection coordinates, A_{setup} gives the setup coefficient, A_{witness} gives the witness coordinate that it multiplies, and the factors u_t include the row-batching weights and the ring-reduction powers. This is a paired affine evaluation. Coefficient bits that are aligned in every contribution can be factored out first; differing ring dimensions leave extra coefficient coordinates with geometric weights.

For sliced $\mathbf{B}$, intersect each slice with the relevant interval of source blocks and use the resulting offset and length; slices that reach the same matrix coefficient add. The $\mathbf{A}$ and $\mathbf{D}$ families are unsliced. Summing the families gives the setup weight without reading any setup coefficient. The setup value itself is authenticated by the setup opening passed to the next level, so no evaluator here assumes that the pseudorandom setup has a small tensor description.

The constant-root analysis of Section 9.4 applies these bounds with controlled factor descriptions and address states. In particular, long equality factors should be given as bit products, while folding challenges given as explicit vectors are charged by their length. For concrete parameters, the verifier may choose among an aligned factorization, a carry evaluation, and direct evaluation of a small factor, counting both preparation and evaluation in each case.

C.5 Verifier Cost of the Batched Row

For the batched row of Section 7.5, the factored blocks cost the equality-weight preparation and carry-state evaluation of their native affine maps (Appendices B.4 and C.3), plus the outer combination. The setup scan performs N_{active} base-field-by-extension-field multiply-adds, plus the transitions needed for the matrix weights and the $\mathbf{B}$-slice sum. Quotient-free checking also computes the transposed kernels, one for each distinct pair of native dimension and witness window; if D_{term} is the sum of their dimensions, this costs $O(D_{\mathsf{term}})$ extension-field operations and storage, independent of the witness length. Admissibility bounds the number of relation-row groups and the transition work, so under those bounds the setup scan is of square-root size whenever the active setup width is. The compression matrices add direct work: evaluating each distinct $\mathbf{F}/\mathbf{H}$ matrix of shape (n_I, m_I, d_I) costs $O(n_I m_I d_I)$ field operations, plus its sparse relation weights, and this work stays with the verifier even at an offloaded level. With quotient lifting, verifier offloading (Section 9) may replace the $\mathbf{A}/\mathbf{B}/\mathbf{D}$ scan by the setup-product sum-check. The admissibility rules require quotient-free folds to use the direct scan; this is a restriction of the schedules, not of Lemma 7.3.

D Deferred Protocol Proofs

D.1 Proofs for the Fold

This subsection proves the shared tensor reduction of Section 5.2 and the norm-check lemmas of Section 6.2.

Proof (Proof of Theorem 5.1). The three terms of the soundness bound come from row batching, claim batching, and the sum-check. If one of the tensor-row identities is false, its discrepancy is a nonzero multilinear polynomial in η, so it vanishes with probability at most $\kappa/|E|$. If the resulting claim-discrepancy vector is nonzero and $N_{\mathsf{tensor}} > 1$, its inner product with the normalized coefficient vector ζ^{tensor} vanishes with probability at most $1/|E|$. The $m_{\max}$-round degree-2 sum-check contributes at most $2m_{\max}/|E|$.

For the honest-abort bound, fix a group g. As a polynomial jointly in the row-batching point η and the common final point ρ, the factor θ_g is nonzero and has degree at most $\kappa + m_{\max}$. Indeed, some Boolean tail point has a nonzero equality factor, and its nonzero base-field coordinate vector gives a nonzero multilinear polynomial in η. The cylindrical factor contributes the remaining final-point degree. Schwartz–Zippel and a union bound over the G groups give the stated bound.

Extraction from the shared tensor reduction. The knowledge-soundness proof extracts from the two batching challenges as follows. For the row-batching point, it uses the nested binary tree of Lemma 10.13 in the κ coordinates of η, which has 2^κ leaves. For the claim-batching vector, varying one of the $N_{\mathsf{tensor}} - 1$ sampled coordinates while fixing the others forces the discrepancy of that claim to vanish, and any accepting equation then forces the first discrepancy to vanish. By Corollary 10.11, this stage therefore has

$$\text{error} \quad \frac{N_{\mathsf{tensor}} - 1}{|E|} \quad \text{and tree factor} \quad N_{\mathsf{tensor}}. \tag{282}$$

Proof (Proof of Lemma 6.3). The accepted digit ranges place the left-hand integer in $[0, U_{\mathsf{dir}}]$, while canonical decoding and the public bound place the right-hand integer in $[0, S_{\max}]$. Both intervals lie in $[0, q)$, so equality of their residues in $\mathbb{F}_q$ is equality in $\mathbb{Z}$.

Proof (Proof of Lemma 6.4). The triangle inequality and the digit bounds give

$$\left| \sum_{u \in I_t} z_a(u) z_{a'}(u) \right| \le |I_t| B_{\mathsf{dig},a} B_{\mathsf{dig},a'} < q/2.$$

There is exactly one integer in $(-q/2, q/2)$ with the claimed residue.

Proof (Proof of Proposition 6.5). For each segment I_t, expand $\sum_{u \in I_t} (\sum_a b^a z_a(u))^2$ and separate the diagonal and off-diagonal digit-plane pairs. Summing the resulting identity over the partition of $[N_A]$ gives (88).

D.2 Proofs for the Ring Checks and Setup Offloading

This subsection proves the ring-check statements of Section 7.3 and the setup-offloading lemma of Section 9.2.

Proof (Proof of Lemma 7.1). (1) is Lemma 3.4 applied blockwise: for each b the lift holds as a polynomial identity in $\mathbb{Z}_q[X]$, and evaluation at $X = \alpha$ is a ring homomorphism, so (116) holds identically in α. *(2)*: suppose block b^* fails over $R_q^{(d_{b^*})}$. For any quotient $\mathbf{r}^{(b^*)}$ with $\deg \mathbf{r}^{(b^*)} < d_{b^*}$, some coordinate of $\mathbf{p}^{(b^*)}(X) := \tilde{\mathsf{M}}^{(b^*)} \tilde{\mathbf{w}}^{(b^*)} - \tilde{\mathbf{h}}^{(b^*)} - (X^{d_{b^*}} + 1)\, \mathbf{r}^{(b^*)}$ is a nonzero polynomial of degree at most $2d_{b^*} - 1 \le 2d_{\max} - 1$ (Section 3.6). The identities (116) can only hold at an α that is a root of that coordinate, of which there are at most $2d_{\max} - 1$; conversely $2d_{\max}$ accepting transcripts with distinct α and the same fixed matrices, witness, targets, and quotients force every coordinate of every $\mathbf{p}^{(b)}$ (each of degree $< 2d_{\max}$) to vanish identically. Reducing modulo $X^{d_b} + 1$ then recovers every block relation, which is the special-soundness claim. *(3)* is (25) restricted to the first $\log d_b$ variables.

Proof (Proof of Lemma 7.3). Multiplication by A in R_d is an F-linear map. Expanding $W = \sum_j w_j X^j$, applying this map term by term, and then applying the F-linear functional $P \mapsto P(\alpha)$ gives

$$\Big(AW \bmod (X^d + 1)\Big)(\alpha) = \sum_j w_j \Big(AX^j \bmod (X^d + 1)\Big)(\alpha).$$

The term on the right is the kernel of (117).

Proof (Proof of Theorem 7.4). For completeness, Lemma 7.3 identifies $z_r(\alpha)$ with the evaluation of the reduced residual Z_r. An honest ring row has $Z_r = 0$ and hence vanishes for every α. The scalar opening residual is zero by hypothesis, so linear row batching preserves zero for the complete residual vector.

For soundness, choose a false ring row r. Its canonical residual $Z_r \in K_r[X]_{<d_r}$ is nonzero, so it has at most $d_r - 1 \le d_{\max} - 1$ roots in E. If α is not a root of Z_r, or if an ordinary field row is false, the evaluated and padded residual vector is nonzero. Its multilinear extension in the s_{rel} coordinates of τ_1 is a nonzero polynomial of total degree at most s_{rel}. Schwartz–Zippel and conditioning on α give (122). Finally, $d_{\max}$ evaluations at distinct values of α determine every degree-$< d_{\max}$ reduced residual.

Proof (Proof of Lemma 9.1). Correct preprocessing commits $C_{S,h}$ to the canonical padded source $\mathbf{S}^\flat$ and fixes its canonical algebraic decommitment state. Compare the source certificate extracted from the accepted successor opening with the canonical reference certificate (162). By Corollary 10.25, either their decoded source vectors agree or their difference yields a matrix collision in $C_{S,h}^{\mathsf{cmp}}$ at its recorded mixed radius. In the former case the successor's source value at ρ_S is $\widetilde{\mathbf{S}^\flat}(\rho_S)$, and soundness of Stage 3 fixes σ_S to the inner product of that source with ω_{setup}. The weight is zero outside the active range $p < N_{\mathsf{active}}$ and agrees with $\bar{\omega}$ of (126) inside it, so the two sums are identical.

E Proofs for Transcript-Tree Extraction

This appendix proves the two extraction results of Section 10.3, in the notation of that subsubsection. The Fiat–Shamir oracle reads message keys (Lemma 10.14(ii)), so a stage input is reconstructed from its key and the table, and the proofs use the facts (F1) and (F2) of Section 10.3. Every expectation over a table counts the addresses queried by the original execution A^J; reruns enter only the cost recurrences.

E.1 Mixed Transcript Trees

We prove Lemma 10.14.

Proof. Interactive bound. Flat stages use Corollary 10.11. For a nested stage, run scalar tree sampling recursively from the last coordinate: at coordinate u, with the earlier coordinates fixed, collect k_u distinct values whose outputs are complete suffix trees for coordinates $u+1,\dots,a$ and the later rounds. These suffix trees may differ, so the result is a nested tree as in Definition 10.12, not coordinate pairs around a single vector.

Each step charges $(k_u-1)/|S_u|$ and multiplies the continuation-tree size by k_u. The charge comes from pruning a product set from its last coordinate upward: a prefix left with fewer than k_u children loses at most $(k_u-1)/|S_u|$ of the uniform conditional mass. A union bound over the coordinates gives e_r, and the weighted scalar sampling argument, applied recursively, gives the product of the k_u in the cost, with polynomial sampling overhead.

Original indices. For the random-oracle statement we give the recursion underlying [43, Section 8, Lemmas 8.2–8.3]. Fix the prover's random tape, let J be the oracle table, whose entries are complete answers, and let $I_r(J) := \langle \xi_r(J) \rangle$ be the key that the original execution selects for stage r, or a failure symbol if it does not reach stage r. Every recursive procedure returns this original index, even when its suffix extraction fails, and never an index selected on a rerun. A successful output also records the original stage input $\xi_r(J)$ and a tree rooted there. The stage-r procedure and its continuations replace only entries of stage r or later, so by (F1) a rerun that selects the key $I_r(J)$ reconstructs the input $\xi_r(J)$, and requiring that input adds no rejection event.

Fixed tapes. Fix the suffix procedure's auxiliary sampling tapes independently of J, keep fresh randomness for the enclosing scalar game, and average over the tapes after applying the sampling bounds. The suffix procedure is then a deterministic table map $M(J) = (V, I_r(J))$, with a tree attached when $V = 1$. For a candidate table, the original execution runs first to determine I_r and its prefix. An index or prefix mismatch is a cheap failure, and only a match invokes the costly suffix procedure, as the cost split of the weighted sampling game requires. At every stage the recursion programs only the selected table entry and keeps all others, including the entries in the message subtree of the selected key. By (F1), each child that it keeps therefore has its original stage input. Within a vector, successive suffix procedures fix the prescribed coordinate prefix and may choose different remaining coordinates.

Representatives. The scalar games run over the values of one challenge coordinate, whereas a table entry is a complete oracle answer, which may also contain rejected draws, encoding bits, and later coordinates. Fix the earlier coordinates of the entry, and let p_d be the probability that its next coordinate takes the value $d \in S$, where $p_d \le 1/|S|$. Before the game, attach to every $d \in S$ a *representative*: with probability $|S|p_d$, a complete answer drawn conditionally on the earlier coordinates and on the value d, and otherwise one drawn conditionally on the earlier coordinates and on a value outside S, which the verifier rejects. The representative of a uniform label then has the law of the entry. The original entry is therefore the representative of a label drawn from its conditional law given the entry: its own value, or, if the entry is rejected, a value d with probability proportional to $1-|S|p_d$. The other representatives are drawn when the game first tries their label. Conditioned on the representatives and the tapes, the game is the finite scalar game over S, so its success and cost bounds apply, and averaging over the representatives gives them for the entry. Here every coordinate is uniform on its set, so $|S|p_d = 1$ and a representative is a conditional answer; the other case arises at the indexed stages of Theorem 10.17.

Original-index support. For an entry i, write J_{-i} for the other entries and define

$$A^0_{r,i}(J_{-i}) := \{z : I_r(J_{-i}, z) = i\}, \qquad P_{0,r} := \sum_i \Pr[A^0_{r,i}(J_{-i}) \neq \varnothing].$$

Every successful recursive output with index i lies in this support, so every support quantity in the recursive sampling games is at most $P_{0,r}$. Let the verifier query every challenge address of the output transcript before it decides, which changes no decision, and let $N_r(J)$ count the stage-r keys queried by the original execution, prover and verifier together. Then

$$P_{0,r} \le \mathbb{E}_J[N_r(J)] \le \mathbb{E}[Q_r] + 1 \le Q + 1, \tag{283}$$

where Q_r counts the prover's stage-r queries. Indeed, if $z \in A^0_{r,i}(J_{-i})$, the execution on (J_{-i}, z) selects i, so its verifier queries i. By (F2), it agrees with the execution on J until their first query to i, so the execution on J queries i as well. This holds for any fixed auxiliary tapes, since the support is defined by the original execution, not by the extractor's reruns.

Coordinates of a nested stage. Let s_{u+1} and C_{u+1} be the success probability and expected cost before coordinate u of a nested stage is added. The scalar game with arity k_u, applied to the suffix map above (which keeps its original index on failure), gives

$$s_u \ge s_{u+1} - P_{0,r}\frac{k_u - 1}{|S_u|}, \qquad C_u \le k_u C_{u+1} + (k_u - 1)P_{0,r}\gamma_u,$$

where γ_u bounds the cheap work of the original execution and of conditional-answer sampling, charged in expectation by conditioning on the sampling tapes and then averaging. Only a matching candidate incurs a suffix extraction, whose complete cost is counted in C_{u+1}; a mismatched candidate costs only cheap work. Telescoping the first recurrence adds the coordinate errors; iterating the second multiplies continuation costs by $\prod_u k_u$, with polynomial additive overhead. The prover receives the whole vector in every run, so there are no virtual protocol messages or extra adversarial oracle queries.

Stages. Induct backward over the protocol stages. The base procedure checks an accepting transcript. At stage r, the continuation returns complete trees for all later stages and keeps the original I_r and stage input even on failure. Later-stage programming occurs only within these continuations and replaces only entries of later stages, so by (F1) every child that a continuation keeps has the original inputs of all earlier stages. A flat stage uses the flat weighted game, and a nested stage the coordinate recursion. With s_{r+1}, C_{r+1} for the continuation,

$$s_r \ge s_{r+1} - P_{0,r}\, e_r \ge s_{r+1} - (Q+1)e_r, \qquad C_r \le b_r C_{r+1} + (b_r - 1)(Q+1)\gamma_r$$

for a polynomial bound γ_r on the stage's cheap work, which also covers rejecting early a rerun that selects a different key.

The success losses therefore add over stages, with no new factor $Q + 1$ at each depth, and the cost is polynomial in Q, the protocol description, and $\prod_r b_r$. The assumed deterministic tree extractor completes the proof. In the interactive case one fixed challenge node replaces the adversarially selected oracle input, so the same recurrences hold with $P_{0,r} = 1$.

E.2 Checkpoint Extraction

We prove Theorem 10.17.

Proof. The extractor is a recursion with one procedure E_r per stage; we call E_r a *node* and its calls to E_{r+1} its *children*. We bound its success probability and expected number of runs by recurrences in r. Plain and ordinary extraction stages reuse Lemma 10.14; indexed extraction stages and checkpoints need separate arguments.

Fixed environment. Fix aux and the random tape of P^*. The extractor uses aux only through the maps of the structure and the reruns of P^*, and never reprograms $\mathcal{O}$, so aux is constant throughout. We prove, for every fixed $\mathsf{aux} \in \mathsf{Good}$,

$$\Pr_J[A^J \text{ accepts and } \mathcal{E} \text{ fails}] \le \mathbb{E}_J\Big[\sum_{a \in \mathsf{Addr}(J)} \epsilon(a)\Big]. \tag{284}$$

Bounding the failure probability by one outside Good and averaging over aux and the tape gives the first inequality of (160). The extractor does not decide Good: its maps are defined for every aux, and only their guarantees use Good.

Procedures. Each E_r, $1 \le r \le m+1$, returns a triple (I, v, y) of an *index tuple* I, a decision bit v, and an output y, omitted when $v = 0$. Elements of $\mathcal{T}$ are tagged outputs without a transcript; a node that receives one from a successful child returns it as its own output, with $v = 1$.

- $E_{m+1}(J)$ runs A^J and returns $I = (\langle \xi_r(J) \rangle)_r$, whose entries are the original indices $I_r(J)$ of Lemma 10.14, the verifier's decision v, and the transcript y.
- For $r \in N$, $E_r := E_{r+1}$.
- For $r \in X$, E_r runs the flat or nested sampling game of stage r from the proof of Lemma 10.14 with continuation E_{r+1}, then applies Comb_r to the collected children. The game's test of whether a resampled table selects the same stage-r key is the first run of A inside E_{r+1}, so it needs no separate run.

For $\ell \in H$, $E_\ell(J)$ runs the following steps.

1. Run $(I, v, y) \leftarrow E_{\ell+1}(J)$. If $v = 0$, return $(I, 0)$.
2. If $y \in \mathcal{T}$ or $\mathsf{Good}_\ell(y) = 1$, return $(I, 1, y)$.
3. Otherwise put $i := \langle \xi_\ell(J) \rangle$ and $w^0 := \mathsf{Cand}_\ell(y)$. Repeat the following until an attempt ends with $v' = 1$ and $\langle \xi_\ell(J') \rangle = i$. Draw a fresh table K' on the message subtree $\mathrm{Sub}(i)$, let J' agree with J outside $\mathrm{Sub}(i)$ and with K' on it, and start $E_{\ell+1}(J')$ with fresh coins. Stop it after its first run of $A^{J'}$ if $\langle \xi_\ell(J') \rangle \neq i$ or $A^{J'}$ rejects; otherwise let it finish with output (I', v', y').
4. If $y' \in \mathcal{T}$, return $(I, 1, y')$. If $\mathsf{Cand}_\ell(y') \neq w^0$, return $(I, 1, \mathsf{Coll}_\ell(y, y'))$. Otherwise return $(I, 0)$.

The extractor runs E_1 with a lazily sampled table and outputs $\mathsf{Wit}(y)$ or the element of $\mathcal{T}$. Step 3 replaces every entry whose key extends i, including the entry at i and every coordinate stream of a later indexed stage, whereas an extraction stage replaces one entry. The set $\mathrm{Sub}(i)$ contains no address of a stage before ℓ, so by (F1) every attempt with key i has the stage-ℓ input $\xi_\ell(J)$. Step 4 never passes a failing candidate to the node above.

Invariants. By backward induction on r, every output (I, v, y) of $E_r(J)$ satisfies the following.

(1) I is the index tuple of A^J, whether or not $v = 1$.
(2) If $v = 1$, then $y \in Y_r^+$; if moreover $y \notin \mathcal{T}$, the transcript recorded in y agrees with that of A^J through the next stage in X.
(3) If $v = 1$, then A^J accepts.

For $r \in X$ they follow from the totality of Comb_r and the construction in the proof of Lemma 10.14, which returns the original index also on failure and keeps a child only when its stage-r key matches; by (F1) such a child has the original stage-r input. For $\ell \in H$, step 2 returns a certificate with $\mathsf{Good}_\ell = 1$ or an element of $\mathcal{T}$, and step 4 applies Coll_ℓ to $y, y' \in Y_{\ell+1}$ with the common stage-ℓ input $\xi_\ell(J)$. Invariant (3) holds because every node first requires its original continuation to succeed. It makes the early stop in step 3 harmless: an attempt stopped because $A^{J'}$ rejects could not have succeeded.

Tapes. The recurrences of Lemma 10.14 require the continuation to be a deterministic map of the oracle table, whereas E_{r+1} uses fresh coins and, at its checkpoints, fresh tables. As in that proof, we fix the tapes before applying the sampling bounds and average over them afterwards, now with tapes that depend on the table: the invocation on table J uses $F(J)$ for a function F from tables to tapes, and we average over a uniform F at the end. For fixed F, E_{r+1} is a deterministic map from tables to a success bit, a cost, and an index tuple satisfying invariant (1); the tables K' drawn at its checkpoints are part of its tape. An enclosing extraction stage samples without replacement, so its invocations visit distinct tables and receive independent tapes.

Plain and ordinary extraction stages. Let s_r be the probability, over a uniform table and the coins of E_r, that E_r returns $v = 1$, and let C_r be its expected number of runs of A. Let $N_r(J)$ and Q_r be as in (283). For $r \in N$, $s_r = s_{r+1}$ and $C_r = C_{r+1}$. For an ordinary stage $r \in X \setminus X_{\mathsf{idx}}$, the proof of Lemma 10.14 uses only the properties of the continuation listed under *Tapes*, and its cheap work per resampled table is one run of A. With (283) it gives

$$s_r \ge s_{r+1} - \varepsilon_r \, \mathbb{E}[N_r(J)], \qquad C_r \le b_r C_{r+1} + (b_r - 1)(Q + 1).$$

Indexed extraction stages. Let $r \in X_{\mathsf{idx}}$. Its flat structure is a reference continuation together with, for each $u \in U_r$, one continuation whose answer differs from the reference only in coordinate u. Each coordinate is a separate table entry, so for each u in turn E_r runs the binary scalar game of Lemma 10.14 on the single entry $a = (i, u)$, where $i = \langle \xi_r(J) \rangle$, with every other entry of J fixed. By (F1), the continuations that it keeps share the stage-r input $\xi_r(J)$. For $a = (i, u)$ put

$$A_a^0(J_{-a}) := \{z : \langle \xi_r(J_{-a}, z) \rangle = i\}.$$

Fix J_{-a}, the suffix tapes, and the representatives of the values in $S_{r,u}$ (*Representatives* in Appendix E.1). The game at a fails while the reference succeeds only if exactly one value has a representative with a successful continuation of stage-r key i, and the label of $J(a)$ is that value. That representative lies in $A_a^0(J_{-a})$, and the label is uniform on $S_{r,u}$, so this event has probability at most $\mathbf{1}[A_a^0(J_{-a}) \neq \varnothing]/|S_{r,u}|$. As for (283), $\mathbf{1}[A_a^0(J_{-a}) \neq \varnothing] \leq \mathbf{1}[a \in \mathsf{Addr}(J)]$ for every J that extends J_{-a}, since an execution that selects i queries every coordinate address of i, and by (F2) the execution on J then queries a too. A union bound over u and a sum over the disjoint events $\langle \xi_r(J) \rangle = i$ give

$$s_r \geq s_{r+1} - \mathbb{E}\Big[\sum_{a \in \mathsf{Addr}_r(J)} \frac{1}{|S_{r,u(a)}|} \Big],$$

where $\mathsf{Addr}_r(J)$ is the set of stage-r addresses in $\mathsf{Addr}(J)$ and $u(a)$ is the coordinate of a. This is the flat game of Lemma 10.14 with one entry programmed at a time, so the cost recurrence $C_r \leq b_r C_{r+1} + (b_r - 1)(Q+1)$ is unchanged.

Checkpoint stages. Fix $\ell \in H$. For a stage-ℓ key i, write a table as $J = (c, K)$, where K is its restriction to the message subtree $\mathrm{Sub}(i)$ and c its restriction to the complement. The keys of stages before ℓ lie outside $\mathrm{Sub}(i)$, so by (F1) the input $T_c(i) := T_{(c,K)}(i)$ does not depend on K. For fixed (i, c), with K and the coins of $E_{\ell+1}$ fresh, let (I, v, y) be the output of $E_{\ell+1}(c, K)$ and define the probabilities that a fresh extraction below i succeeds and selects i, that it does so with candidate w, and that this certificate also fails the test:

$$\begin{aligned} P_{i,c} &:= \Pr[v = 1,\ \langle \xi_\ell(c, K) \rangle = i], \\ q_{i,c}(w) &:= \Pr[v = 1,\ \langle \xi_\ell(c, K) \rangle = i,\ y \notin \mathcal{T},\ \mathsf{Cand}_\ell(y) = w], \\ f_{i,c}(w) &:= \Pr[v = 1,\ \langle \xi_\ell(c, K) \rangle = i,\ y \notin \mathcal{T},\ \mathsf{Cand}_\ell(y) = w,\ \mathsf{Good}_\ell(y) = 0]. \end{aligned}$$

For $j \in \mathrm{seg}(\ell)$, let $N_{i,j}(c, K)$ be the number of stage-j keys in $\mathrm{Sub}(i)$ that $A^{(c,K)}$ queries, counting the queries of P^* and of the verifier. Since i is the only stage-ℓ key in $\mathrm{Sub}(i)$, $N_{i,\ell} \in \{0, 1\}$.

Claim 1. If $D(T_c(i); w) = 1$, then

$$q_{i,c}(w) \leq \sum_{j \in \mathrm{seg}(\ell)} \varepsilon_j \, \mathbb{E}_K[N_{i,j}(c, K)].$$

Suppose $E_{\ell+1}(c, K)$ returns $y \in Y_{\ell+1}$ with candidate w and stage-ℓ key i, and put $J = (c, K)$. By invariant (2), the stage inputs $\xi_\ell, \ldots, \xi_{\bar{\ell}+1}$ of y are those of A^J, so $\xi_j = T_J(k_j)$ for the keys $k_j := \langle \xi_j(J) \rangle \in \mathrm{Sub}(i)$, the challenges are $K(k_j)$, and $\xi_\ell = T_c(i)$. Condition (c) gives $D(\xi_{\bar{\ell}+1}; w) = 0$, while $D(\xi_\ell; w) = 1$, so by condition (b) some stage $j \in \mathrm{seg}(\ell)$ has $D(\xi_j; w) = 1$ and $K(k_j) \in \mathsf{Bad}_j(\xi_j; w)$. The verifier queries k_j. Hence $q_{i,c}(w)$ is at most the sum, over $j \in \mathrm{seg}(\ell)$ and the stage-j keys $k \in \mathrm{Sub}(i)$, of

$$\Pr_K\big[A^J \text{ queries } k,\ D(T_J(k); w) = 1,\ K(k) \in \mathsf{Bad}_j(T_J(k); w)\big],$$

an event of the original execution alone. Fix k and condition on every entry of K except $K(k)$. By (F1), $T_J(k)$ is then fixed, since it reads only entries at keys of earlier stages; by (F2), so is the event that A^J queries k; and $K(k)$ is still distributed as Ω_j. The term is therefore at most $\varepsilon_j \Pr_K[A^J \text{ queries } k]$, and summing over k gives the claim. The bound holds for every fixed c and every query order. In particular, A may query k before the keys from which $T_J(k)$ is reconstructed, since the conditioning fixes those entries without revealing them along the run. The case $j = \ell$ is the key $k = i$, with $T_J(i) = T_c(i)$.

Claim 2. In both bounds below, i ranges over the stage-ℓ keys. For every $j \in \mathrm{seg}(\ell)$ and every table J,

$$\sum_i N_{i,j}(J) = N_j(J) \leq Q_j + 1.$$

Moreover, if c_i is the restriction of a uniform table J to the complement of $\mathrm{Sub}(i)$, then

$$\sum_i \Pr_{c_i}[P_{i,c_i} > 0] \leq Q + 1.$$

A stage-j key k with $j \geq \ell$ extends exactly one stage-ℓ key, its truncation $k_{\leq \ell}$, so it lies in exactly one set $\mathrm{Sub}(i)$. This gives the equality; the verifier adds at most one stage-j key to those of P^*. For the second

bound, fix J. If P^* makes no query in $\mathrm{Sub}(i)$, replacing the table there does not change its run, so i can be the stage-ℓ key after the replacement only if $i = \langle \xi_\ell(J) \rangle$. Hence $P_{i,c_i(J)} > 0$ for at most one key i besides those whose set $\mathrm{Sub}(i)$ receives a query of P^*, and these sets are disjoint, so at most Q of them receive one; taking the expectation over J gives the bound. It cannot be improved to $\mathbb{E}[Q_\ell] + 1$: a prover may query later-stage keys in several sets $\mathrm{Sub}(i)$ without querying any stage-ℓ key, and let their answers decide which i its proof uses.

Success. E_ℓ returns $v = 1$ whenever step 1 does, except in the event Fail_ℓ that step 1 returns y with $\mathsf{Good}_\ell(y) = 0$ and candidate w^0, and the loop ends with $y' \notin \mathcal{T}$ and $\mathsf{Cand}_\ell(y') = w^0$. By invariant (2) and (F1), $\xi_\ell(y) = T_{c_i}(i)$, so condition (a) gives $D(T_{c_i}(i); w^0) = 1$. Given c_i, the attempts of step 3 are independent copies of $E_{\ell+1}(c_i, K')$, independent of the first run, and the loop stops at the first with $v' = 1$ and key i; by invariant (3) the early stop discards no such copy. Its output therefore has candidate w with probability $q_{i,c_i}(w)/P_{i,c_i}$, and

$$\begin{aligned}\Pr[\mathsf{Fail}_\ell] &= \sum_i \mathbb{E}_{c_i}\Big[\sum_{w : D(T_{c_i}(i); w) = 1} f_{i,c_i}(w) \frac{q_{i,c_i}(w)}{P_{i,c_i}} \Big] \\ &\le \sum_i \mathbb{E}_{c_i}\Big[\max_{w : D(T_{c_i}(i); w) = 1} q_{i,c_i}(w) \Big],\end{aligned}$$

with no contribution when $P_{i,c_i} = 0$, since $\sum_w f_{i,c}(w) \le P_{i,c}$.

For each i these are joint probabilities that include the event $\langle \xi_\ell \rangle = i$; nothing is conditioned on the selected key, and each expectation is over the marginal law of its own c_i. By Claim 1 the maximum is at most $\sum_{j \in \mathrm{seg}(\ell)} \varepsilon_j \, \mathbb{E}_K[N_{i,j}(c_i, K)]$, and (c_i, K) is a uniform table. The first part of Claim 2 now gives

$$s_\ell \ge s_{\ell+1} - \Pr[\mathsf{Fail}_\ell] \ge s_{\ell+1} - \sum_{j \in \mathrm{seg}(\ell)} \varepsilon_j \, \mathbb{E}[N_j(J)].$$

Cost. For fixed (i, c), step 3 is entered with selected key i with probability at most $P_{i,c}$ and then makes $1/P_{i,c}$ attempts in expectation. Define the unnormalised expectation

$$\Gamma_{i,c} := \mathbb{E}_K\big[\mathrm{cost}(E_{\ell+1}(c, K)) \, \mathbf{1}[\langle \xi_\ell(c, K) \rangle = i]\big].$$

One attempt costs at most $1 + \Gamma_{i,c}$ in expectation. By Wald's identity, the second part of Claim 2, and $\sum_i \mathbb{E}_{c_i} \Gamma_{i,c_i} \le C_{\ell+1}$, the expected cost of step 3 is at most

$$\sum_i \mathbb{E}_{c_i}\big[\mathbf{1}[P_{i,c_i} > 0](1 + \Gamma_{i,c_i})\big] \le (Q + 1) + C_{\ell+1}.$$

Hence $C_\ell \le 2C_{\ell+1} + (Q + 1)$.

Conclusion. Let $\varepsilon := \Pr_J[A^J \text{ accepts}] = s_{m+1}$. By invariant (3), success implies acceptance, so the extractor fails on an accepting execution with probability $\varepsilon - s_1$. Telescoping the success recurrences gives

$$\varepsilon - s_1 \le \mathbb{E}\Big[\sum_{r \notin X_{\mathsf{idx}}} \varepsilon_r N_r(J) + \sum_{r \in X_{\mathsf{idx}}} \sum_{a \in \mathsf{Addr}_r(J)} \frac{1}{|S_{r,u(a)}|} \Big] = \mathbb{E}\Big[\sum_{a \in \mathsf{Addr}(J)} \epsilon(a) \Big],$$

which is (284). The error of each plain stage appears once because the stage lies in exactly one segment. For the second inequality of (160), the addresses of the output transcript have total weight at most $\varepsilon_{\mathsf{ck}}$, since the coordinate weights of an indexed stage sum to e_r, and P^* queries at most Q further addresses, each of weight at most $\epsilon_{\max}$.

The cost recurrences have the common form $C_r \le kC_{r+1} + (k-1)(Q+1)$, with $k = 1$ at a plain stage, $k = 2$ at a checkpoint, and $k = b_r$ at an extraction stage. With $C_{m+1} = 1$ this solves to $\widehat{K} + (Q+1)(\widehat{K} - 1)$. Each run and its attached work cost at most T_{run} in expectation given the history before it, so the expected cost is at most this number times T_{run}. As in Lemma 10.14, the interactive protocol has one fixed challenge node per stage, so the recurrences hold with $Q = 0$.

F Constant-Root Instantiation

This appendix proves Theorem 9.3 and Lemma 9.4, and discusses which hardness assumptions imply the parameter-growth assumption of Section 9.4.

Proof (Proof of Theorem 9.3). Early folds. By (143), these folds leave a witness of size $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N_\star)$, after which the core recursion continues to the terminal check of Section 8.2. At each early fold, the verifier processes the B_h challenge-dependent contributions in $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N_\star)$ time, and setup offloading removes the only scan whose width can be M_h, so the verifier needs no expanded $\mathbf{A}/\mathbf{B}/\mathbf{D}$ prefixes. It still evaluates the $\mathbf{F}/\mathbf{H}$ maps: the first $\mathbf{F}$ map acts on digits of the stacked $\mathbf{B}$ image ($S_B n_B d_B$ coefficients), the first $\mathbf{H}$ map on digits of the $\mathbf{D}$ image ($n_D d_D$ coefficients; Section 4.5), and each later map on digits of the preceding compression image. By (R1) and (R2), this work and its expanded storage are $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(1)$, so these maps need no additional regularity condition. The early folds add a constant number of logarithmic transcripts.

Direct suffix. The suffix begins at total source size $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N_\star)$, so the active matrix prefix of each direct fold obeys the same bound. The parameter bounds and contraction of the suffix give

$$\sum_h N_h = \widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N_\star) \qquad \text{and} \qquad \sum_h \log N_h = O(\log N)$$

over the suffix. The first sum bounds its verifier work and expanded setup storage, and the second its transcripts under condition (iii). The terminal adds only polylogarithmic cost.

Prover. The witness sizes at successive levels form a geometric series beginning at N. Summing the total source lengths, including carried setup groups, gives the prover bound.

Proof (Proof of Lemma 9.4). Parameter choices. We use:
- one root opening, a common ring degree d for all maps, module rank one, and one $\mathbf{B}$ slice;
- $E = \mathbb{F}_q$ (extension degree one), the entire ring as the challenge subring at the first two levels, and the corresponding evaluation-trace encoding later;
- quotient lifting throughout the offloaded prefix and while checking its final setup opening (a later direct suffix may use quotient-free checks);
- the balanced digit base $b = b_1 = 4$, without Euclidean norm certificates, operator filtering, or response-dependent grinding;
- two-stage binary compression of the root and setup-prefix commitments, with outputs not necessarily of the concrete 128-byte size, through the offloaded prefix and the fold that receives its last setup group, and uncompressed commitments in the direct suffix, so the schedule switches from compressed to uncompressed commitments once;
- enough setup-sampling bits per coefficient for negligible total sampling bias under Section 4.3.

Quotient lifting is chosen because its factored weights simplify the cost analysis, not because offloading requires it. Every ring and digit conversion has $L^{O(1)}$ overhead: full-field source coefficients need $O(\log q)$ base-four digits, and quotient coefficients have bit length $O(\log m + \log q + \log d)$, so their digit depths are polynomial in L.

Challenges. Use all sign vectors with exactly $d/2$ nonzero entries: there are $\binom{d}{d/2}2^{d/2} = 2^{\Omega(d)}$ of them, their coefficient one-norm is $d/2$, and a difference has coefficient bound two. Since $q \equiv 5 \pmod 8$, the power-of-two cyclotomic ring has two irreducible factors, so the invertibility condition of Section 3.2 holds. Increasing C in $d \geq CL$ covers the sum of the folding errors and the factor $Q_{\max} + 1$; the modulus covers all field-challenge errors. A folded digit vector with B_h blocks has deterministic coefficient bound $O(B_h d)$, hence $O(\log B_h + \log d)$ base-four planes, so neither honest completeness nor the challenge count requires a response model or an operator filter.

To sample the base family exactly, draw a uniform integer index into it by binary rejection and unrank its support and signs in time polynomial in d. Each trial succeeds with probability at least $1/2$, so CL trials per coordinate give a fixed sampler tape failing with probability at most 2^{-CL}; a union bound over the coordinates covers sampler failure within the completeness budget, and the accepted distribution is uniform. All commitment collision bounds and matrix widths are $2^{O(L)}$, so the assumed SIS bounds apply once C is large enough.

Early folds. At each early fold, choose for each of the at most two source groups a power-of-two block length within a factor two of the group's length divided by $N_\star$, and at least one ring position. Each group has

$O(N_\star)$ blocks, and all response planes together have size $L^{O(1)}N_h/N_\star$. Inner images, quotient polynomials, compression digits, and other auxiliary digits have total size $L^{O(1)}N_\star$, since ranks, ring degrees, and claim counts are polynomial in L and there are constantly many source groups. Let $|\mathbf{w}^{(h+1)}|_K$ be the length of this successor witness. By (140), the setup prefix, including padding, has length $N_{\mathsf{prefix},h} \le 2|\mathbf{w}^{(h+1)}|_K$. The next fold takes both as source groups, each split by the same rule. Thus

$$N_{h+1} = |\mathbf{w}^{(h+1)}|_K + N_{\mathsf{prefix},h} \le 3|\mathbf{w}^{(h+1)}|_K \le L^{O(1)}(N_h/N_\star + N_\star),$$

and after $\kappa_{\mathsf{root}} - 1$ early folds the remaining size is $L^{O_{\kappa_{\mathsf{root}}}(1)}N_\star$. Balanced direct folds follow, down to witness size $L^{O(1)}$, where the terminal check applies; inputs already below this threshold use a direct schedule from the start. A large enough threshold absorbs the digit and ring overhead in each direct fold.

Regularity. The common ring dimension and consecutive digit layout leave $L^{O(1)}$ coordinate ranges and relation families, each with a constant number of simultaneous affine addresses. After peeling the power-of-two position strides, the remaining strides and carry ranges involve only ring lanes and digit indices, so they are polynomial in L. Long position axes enter through equality-bit factors; the only long explicit local factors are the $O(N_\star)$ folding challenges. The carry-peeling and paired contractions of Appendices B and C.4 therefore evaluate the remaining weights in $L^{O(1)}N_\star$ operations without scanning a long position interval. Each active two-stage binary compression map acts on only $L^{O(1)}$ coefficients (a rank-one image has d coefficients, and its binary expansion $d\lceil \log_2 q \rceil$ digits), so its work obeys the same bound. These choices verify (R1)–(R5).

Direct suffix and extraction tree. Fixed digit bases give constant sum-check degree. Above a sufficiently large polynomial-in-L stopping threshold, each balanced direct fold reduces N_h to at most $N_h^{3/4}$. Consequently $\sum_h \log N_h = O(\log N)$ over the suffix, which has $O(1+\log(\log N/\log L))$ folds, so their accumulated $O(\log L)$ costs are also $O(\log N)$. Each fold has $O(\log N_h + \log L)$ constant-degree rounds and polynomially many claim-combination branches in N_h and L, and the fixed early prefix and the terminal add only $O_{\kappa_{\mathsf{root}}}(\log N + \log L)$ to the logarithm of the extraction-tree size. Thus the complete tree, and with it the smaller tree of (197), has size $(NL)^{O_{\kappa_{\mathsf{root}}}(1)}$, as admissibility requires; the exact security allocation uses Equations (142) and (195).

Sufficient hardness assumptions. A suitable exponential hardness guarantee suffices for polynomial parameter growth but is not necessary, and "subexponential hardness" alone is too imprecise to decide the matter. Suppose the relevant SIS instances provide $\lambda_{\mathsf{SIS}}(D)$ bits of hardness at an effective size D, in the success-probability sense of the parameter-growth assumption rather than as the cost of one known attack. Here λ_{SIS} accounts for the modulus, width, collision norm, and reduction losses, and implementing size D costs a polynomial in D and L. We need $\lambda_{\mathsf{SIS}}(D) = \Omega(L)$.

- If $\lambda_{\mathsf{SIS}}(D) = \Omega(D^\alpha)$ for some fixed $\alpha > 0$, including attack-time lower bounds $2^{\Omega(\sqrt{D})}$, choosing $D = L^{O(1/\alpha)}$ preserves every displayed soft-O bound.
- If the guarantee is only $2^{\Omega((\log D)^c)}$ for a fixed $c > 1$, the analogous sufficient choice $D = 2^{O(L^{1/c})}$ has overhead polynomial in D, which is not polylogarithmic in N. For fixed λ and polynomial $Q_{\max}$, this gives prover cost $N^{1+o(1)}$, verifier cost $N^{1/\kappa_{\mathsf{root}}+o(1)}$, and proof size $N^{o(1)}$; such a quasipolynomial guarantee does not by itself justify the stronger soft-O bounds.

G SIS Security Estimation

Akita sizes every matrix through the Module-SIS problem of Definition 3.2. This appendix explains the hardness target, how we estimate attack costs, and how the planner uses those estimates to choose module ranks.

Hardness target. We estimate the cost of attacking each concrete Module-SIS instance in the quantum ADPS16 model, which assigns cost $2^{0.2650\beta}$ to lattice reduction with block size β; the estimator searches over the attack parameters and reports the cheapest attack it finds. For the concrete parameter family studied here, every Module-SIS instance must have estimated cost at least 2^{128}. This per-instance floor applies to outer commitments $\mathbf{B}_g$, opening commitments $\mathbf{D}_{\mathsf{batch}}$, inner commitments $\mathbf{A}$, and every compression map $\mathbf{F}_i, \mathbf{H}_i$. Not every level has every matrix: the terminal level, whose inner state $\mathbf{t}$ is bound by the preceding level, contributes an $\mathbf{A}$ collision instance and no $\mathbf{B}$, $\mathbf{D}$, $\mathbf{F}$, or $\mathbf{H}$ instance.

The security proof composes these instance-wise assumptions. Let c_I be the estimated quantum attack cost, in bits, of instance I. For a public policy $\mathcal{P}$ with accepted schedule family $\mathcal{S}_{\mathcal{P}}$, we report the combined estimate

$$\lambda_{\mathsf{fam}} := -\log_2\left(\sum_{I\in\mathbb{I}_{\mathcal{P}}} 2^{-c_I}\right), \tag{285}$$

where $\mathbb{I}_{\mathcal{P}}$ contains each distinct shared-prefix matrix instance induced by that family, as in Lemma 10.20. The direct audit reported in the artifact gives $\lambda_{\mathsf{fam}} > 125$ quantum bits; every fixed schedule is stronger, the weakest audited one exceeding 127 quantum bits [66]. We state the lower 125-bit figure because it also covers an adversary that chooses a schedule from the entire public family after seeing the shared setup.

We do not regard the gap between 128 quantum bits per instance and 125 bits family-wide as a security concern. The quantum ADPS16 Core-SVP model is already conservative: it charges the attacker only the idealized quantum shortest-vector cost and omits the rest of BKZ. The union bound is conservative in a different direction: it adds the modeled advantages of many heterogeneous matrix instances, but corresponds to no single attack that amortizes work across them. The family-wide loss is therefore proof-level accounting, not evidence of a 2^{125}-work quantum attack on Akita.

Norms and basis shape. Coefficient-ℓ_∞ instances use the direct infinity-norm attack equations with LGSA for the reduced-basis shape. Euclidean instances use the Euclidean attack equations. Both keep the quantum ADPS16 cost model and the same 128-bit per-instance floor; the norm is part of the instance supplied by the reduction, not an estimator-side conversion. LGSA is the L-shaped geometric-series-assumption profile of `simulator.LGSA` in `lattice-estimator`, in which basis rerandomization removes the initial q-vector plateau; ZGSA is the corresponding Z-shaped profile of `simulator.ZGSA`, which allows that plateau. Our Rust implementations use the SIS settings `xi=1` and `tau=False`, with the profile corrections described below [66].

From a lattice problem to a bit-security estimate. A generic attack on an SIS instance searches for a nonzero short vector in the associated q-ary lattice, after first running BKZ to improve the basis. A larger BKZ block size β finds shorter vectors but requires solving a harder shortest-vector problem inside each block. An estimator therefore makes two predictions: how large β must be before the desired collision appears in the reduced basis, and how much work reduction at that block size costs.

The first prediction depends on the collision norm and on the simulated shape of the reduced basis. The second is the *reduction-cost model*, for which ADPS16, BDGL16, and MATZOV are three choices. None changes the SIS problem or the short vector that breaks binding; they change what work is counted once β is determined. Thus "128 bits" does not mean that a theorem forces every attack to perform 2^{128} machine instructions, but that the cheapest attack found by the estimator costs at least 2^{128} work units under the stated basis-shape and reduction-cost models.

ADPS16 counts only the hardest shortest-vector call inside BKZ; BDGL16 counts a classical BKZ run assembled from many such calls; and MATZOV gives a more detailed classical gate-count model for a progressively stronger sequence of reductions. In the high-block-size regime relevant here, the resulting estimates usually satisfy

$$\begin{aligned}\text{quantum ADPS16} &< \text{classical ADPS16}\\ &< \text{classical MATZOV} < \text{classical BDGL16}.\end{aligned}$$

The first inequality reflects the assumed quantum speedup; each later one charges the classical attacker for more of the computation around the shortest-vector call. None of these changes makes the underlying SIS instance harder.

ADPS16: the Core-SVP convention. ADPS16 treats the dominant work in BKZ as one shortest-vector computation in dimension β and maps β directly to an attack cost: $2^{0.2920\beta}$ classically and $2^{0.2650\beta}$ quantumly [5]. At the 128-bit quantum floor the classical figure is about 2^{141}; this is the same floor in classical terms, not a separate acceptance rule. The formulas intentionally omit the cost of preparing the basis, of invoking the shortest-vector routine many times during BKZ, and of managing the required memory, so the output is a Core-SVP yardstick, not a gate count, running time, or qubit count.

We use the quantum convention because it gives a simple post-quantum target and does not credit Akita with the extra cost of a particular classical BKZ implementation. A newer asymptotic quantum sieve lowers the idealized SVP exponent from 0.2650 to 0.2563 by reusing a large quantum state across many collision searches [23]. We do not use it as the acceptance criterion, because transferring it to an end-to-end BKZ attack requires strong assumptions about reusable quantum memory and repeated sieving. We did test it as a sensitivity model: an independently optimized sweep found no accepted ADPS16 row below the correspondingly rescaled threshold.[12]

BDGL16: classical BKZ accounting. BDGL16 starts from a classical lattice sieve, an exponential-time algorithm that stores many lattice vectors and searches for combinations that give shorter ones; its asymptotic shortest-vector cost is again about $2^{0.292\beta}$ [15]. The implementation in `lattice-estimator` adds a concrete offset for the sieve, the cost of an initial LLL basis-reduction pass, and an estimate of the number of shortest-vector calls BKZ makes, modeled as $8d_{\mathsf{lat}}$ calls in the common regime $\beta < d_{\mathsf{lat}}$, where d_{lat} is the lattice dimension. This makes BDGL16 a more detailed classical estimate, though still not a measured running time or a complete implementation cost. By counting work that Core-SVP deliberately omits, it also gives the defender a larger security estimate.

MATZOV: a progressive-sieving cost model. The MATZOV report develops an improved dual attack on LWE and refines the estimated classical cost of lattice sieving [78]. As a reduction-cost model for SIS, `lattice-estimator` uses only the latter refinement, not the report's full LWE attack. The model charges for a progressive BKZ computation, a sequence of increasingly strong reductions, rather than one abstract shortest-vector call. It also models the effective dimension of the final sieve and assigns gate counts to the nearest-neighbor and list-decoding subroutines that search for lattice vectors likely to cancel. Its output therefore depends on β, on the lattice dimension d_{lat}, and on the selected classical nearest-neighbor backend. MATZOV credits the attacker with "dimensions for free", charging the final sieve in an effective dimension below β. This credit usually makes MATZOV cheaper than BDGL16, while its progressive-BKZ accounting keeps it more expensive than the single-call ADPS16 convention. No single MATZOV exponent can replace 0.2650 or 0.2920.

Comparison of reduction-cost models. The ordering above is not a theorem for every β and d_{lat}. ADPS16 depends only on β, whereas MATZOV and BDGL16 also depend on d_{lat}, and their estimates can cross at small block sizes. At the reference point $\beta = 500$ and $d_{\mathsf{lat}} = 1024$, the current `lattice-estimator` formulas give 132.5 bits for quantum ADPS16, 146.0 for classical ADPS16, 169.7 for classical MATZOV, and 175.4 for classical BDGL16 [4,5,15,78]. MATZOV's value reflects a progressive BKZ run with the effective sieve dimension reduced from 500 to about 457.4. These numbers compare only the reduction-cost models: they fix both the SIS instance and the required block size.

End-to-end estimates can also differ one step earlier: a basis-shape model such as GSA or LGSA can predict a different block size β for the same SIS instance, to which ADPS16, BDGL16, or MATZOV then assigns a cost. RoKoKo's released estimator uses GSA followed by classical MATZOV accounting, the Orthus comparison reports both classical ADPS16 and BDGL16, and Akita's acceptance criterion uses LGSA followed by quantum ADPS16 [4,78,5,15]. Their bit counts therefore do not measure the same unit of work.

The scope is narrower still. Akita scalarizes a module matrix of ring dimension d to ordinary SIS dimensions $n_{\mathsf{sc}} = nd$ and $m_{\mathsf{sc}} = md$. None of the models above accounts for attacks that exploit the ring or module structure, CRT decompositions, subfield projections, or role-specific structure in Akita's matrices. We therefore record each artifact's own security model and any normalization separately in Section 13. A different estimator is not by itself an implementation error: Appendix J concerns mismatches between the relation claimed by a protocol and the one enforced by its implementation.

Estimator interface. Each candidate matrix induces an instance $\mathsf{MSIS}^{(p)}_{q,d}(n, m, \eta)$ for $p \in \{2, \infty\}$, where the schedule fixes the field modulus q, ring dimension d, matrix shape (n, m), norm p, and collision bound η. Offline table generation scalarizes this tuple and passes it to our standalone Rust estimator, which follows the methodology and Python/Sage implementation of the public `lattice-estimator`, with matching formulas

[12] The model comparison, assumptions, and sweep are recorded in Akita's SIS policy specification [66].

where the two models agree [4,66]. The generator searches for the cheapest attack under the fixed norm, shape, and reduction-cost model and records the smallest rank that reaches the required floor (Table 11).

The Rust implementation is about three orders of magnitude faster than the Python/Sage version, which makes a complete sweep over the thousands of cells in each production profile practical. Speed does not replace validation: we retain immutable Sage outputs for trusted reference cells and test the Rust implementation against them. Where the reference implementation is not sound for Akita's parameter range, our estimator deliberately defines a corrected revision. The documented corrections cover the centered Gaussian mass at tiny probabilities, the active dimension after the attacker removes coordinates, the valid domain for tall q-ary lattices, and determinant preservation for unbalanced ZGSA profiles. Production LGSA rows also include the ordinary Euclidean attack when it applies, and certify every computed boundary over the complete supported search domain. The estimator revision and generated-table digest bind these choices; agreement with a faster but uncertified local-minimum search is not sufficient.

Collision norms. The collision norm η is read from the fold reduction in Section 10, which covers both single- and multi-instance folds. At a packing fold, the $\mathbf{A}$ bound uses the certified challenge-subring challenge and its embedding into the $\mathbf{A}$ ring; at an evaluation-trace fold, it uses the certified full-ring challenge. The tensor-reduction prefix contributes to the Fiat–Shamir error budget, but it does not create another matrix or let the estimator reuse a packing challenge bound for an evaluation-trace instance. For the $\mathbf{B}$ and $\mathbf{D}$ matrices, the transcript-difference bound is $\eta = b - 1$ on the certified balanced base-b alphabet of the level that range-checks the digits (Remark 10.24); the honest digits are base-b_1 with $b_1 \le b$. For a coefficient-ℓ_∞ $\mathbf{A}$ matrix, the bound is $2\bar{\kappa}_1\bar{\beta}_\infty$, with $\bar{\kappa}_1 \le 2\omega$ and $\bar{\beta}_\infty$ the certified fold-response-difference envelope of Theorem 10.31. For a Euclidean $\mathbf{A}$ matrix, the reduction supplies one of the three bounds in (177). A challenge operator bound Γ and a folded-response norm certificate $\|\mathbf{z}\|_2^2 \le S$ give the squared collision bound $\eta_A^2 = 64\Gamma^2 S$. The same formula may instead use the ℓ_1 challenge radius, or reverse the multiplication and use the ℓ_2 challenge radius with the response multiplication-operator norm. These are three bounds on the same kernel vector, so the planner may select any independently certified one.

All response quantities in these formulas are norms of centered ring coefficients. The logical-to-ring embedding factor is applied only when the certified source quantity is stated before packing, and never again to a folded response or its reconstructed base-field digit planes.

Euclidean table units and lookup. For an $\mathbf{A}$ matrix of module rank r, ring dimension d, and input width m_A, the Euclidean estimator receives the scalar dimensions

$$n_{\mathsf{sc}} = rd, \qquad m_{\mathsf{sc}} = m_A d, \qquad L_{\mathsf{sc}} = \sqrt{C_2}, \tag{286}$$

where C_2 is the squared norm bound for the complete scalar collision vector. For the response-norm route, $C_2 = 64\Gamma^2 S_{\max}$ by Corollary 10.35. The bound $S_{\max}$ already sums the squares of all $m_A d$ input coefficients, so the input width appears only in m_{sc}; multiplying C_2 by m_A or d would count those coordinates twice.

The Euclidean table is keyed by the squared quantity C_2, not by its square root. Given a bound $C_2 > 0$, lookup uses the least supported power-of-two bucket $\widehat{C}_2 \ge C_2$. For each modulus profile, ring dimension, module rank, and bucket, a row records the largest supported ring width m_A, and the planner chooses the first rank whose recorded width is at least the requested m_A. Rounding the bucket up is sound because it asks the estimator to secure the larger radius $\sqrt{\widehat{C}_2}$. If the table has no supported bucket, dimension, or rank, the Euclidean route is unavailable and the planner must retain another certified route.

Table generation. For coefficient-ℓ_∞ instances, offline generation computes scalar cutoffs

$$(\text{modulus profile}, \eta, n_{\mathsf{sc}}) \longmapsto \max m_{\mathsf{sc}}.$$

The generator expands the matrix roles into every reachable ring dimension, collision bound, and rank, then deduplicates cells that induce the same scalar instance. For each computed cutoff it records both the accepted boundary and its immediate rejected successor. The compact runtime projection is keyed by the policy, modulus profile, matrix role, ring dimension, and collision bound; for each rank it returns the largest supported ring width. The separate Euclidean table stores maximum-width rows keyed by $(q, d, r, \widehat{C}_2)$, using the scalar conversion in (286).

Modulus q	d_1	m_1 (max.)	d_2	m_2	intermediate	output
$2^{32} - 99$	64	1024	32	64	256 B	128 B
$2^{64} - 59$	32	2048	16	128	256 B	128 B
$2^{128} - 2^{32} + 22537$	16	4096	8	256	256 B	128 B

Table 10: Two-stage binary compression for input images of at most 8 KiB. Both maps have module rank one and coefficient collision bound one. Widths count ring elements; m_1 is evaluated at the maximum input size. Security estimates use the quantum ADPS16 Core-SVP model (Appendix G).

Commitment compression (Section 4.5) adds one estimator query for each map of each two-stage pair. Each map is estimated at its own ring dimension as $\mathsf{MSIS}_{q,d_j}(n_j, m_j, 1)$, where m_j counts ring elements in its complete binary digit vector, including padding. The fused binary constraint covers every coefficient of that vector, so two accepted openings differ coefficientwise by at most one. Both maps must meet the selected security target; the concrete rank-one choices appear in Table 10. The original $\mathbf{B}, \mathbf{D}$ matrices keep their own MSIS floors, since the extraction of Lemma 4.3 may still end in a collision for them.

Choosing the compression output size. The two-stage construction does not prescribe a fixed number of output bytes. For a source with s_0 field coefficients, let the two maps have ring dimensions d_1, d_2 and module ranks n_1, n_2, and write $\delta = \lceil \log_2 q \rceil$. Decomposing the source and then the first image gives input widths

$$m_1 = \left\lceil \frac{\delta s_0}{d_1} \right\rceil, \qquad m_2 = \left\lceil \frac{\delta n_1 d_1}{d_2} \right\rceil.$$

The intermediate image contains $n_1 d_1$ coefficients, and the public output contains $n_2 d_2$ coefficients. We choose both shapes so that $\mathsf{MSIS}_{q,d_i}(n_i, m_i, 1)$ meets the selected security target for $i = 1, 2$. With $\lceil \delta/8 \rceil$ bytes per field element, the output occupies $n_2 d_2 \lceil \delta/8 \rceil$ bytes. Larger inputs or a higher security target can require larger outputs; the two stages and their binary alphabet remain unchanged. For the concrete parameters in Table 10, each complete input image of at most 8 KiB produces a 256-byte intermediate image and a 128-byte public output.

Concrete compression estimates. For the parameters of Table 10 at the maximum source size of 8 KiB, scalarization gives each first-stage instance 65536 input coordinates and each second-stage instance 2048, with output dimensions 64, 32, 16 and 32, 16, 8 for the 32-, 64-, and 128-bit moduli. Exhaustive search over the block size and the effective lattice dimension, using the LGSA basis-shape model and quantum ADPS16 Core-SVP cost, gives 147.34 bits for each first-stage instance and 153.17 bits for each second-stage instance, at minimizing block sizes 556 and 578 through the cost 0.2650β. Smaller supported source images use prefixes of the same first-stage maps, and zero-padding a collision embeds it into the maximum-width instance. Thus every compression instance in this parameter family has at least 147 bits of estimated quantum security under this model. These per-map estimates enter the combined accounting in (285) alongside the other commitment instances.

Using the estimates in parameter selection. The planner uses these estimates to select candidate ranks, and independent schedule validation checks the resulting instances and their combined accounting before the parameters are used for proving and verification (Section 12.5). Concrete tables and their generation rules are documented with the parameter-selection artifact [66,65].

Table 11: Inputs and output of the Module-SIS security table.

Parameter	Type	Meaning
q	Input	Field modulus
d	Input	Ring dimension
m	Input	Matrix column count
p	Input	Norm of the collision, 2 or ∞
η	Input	Certified collision bound
n	Output	Least module rank meeting the security floor

H Certifying Operator-Norm Challenge Filters

Akita uses operator-norm rejection sampling for sparse folding challenges in some Euclidean security profiles, as in (23). This requires two guarantees. First, the integer predicate replayed by the verifier must never accept a challenge whose true multiplication norm exceeds the threshold. Second, its accepted set must remain large enough for the security analysis. The first follows from a fixed-point enclosure of every spectral value (Theorem H.1), the second from exact spectral moments and a rational polynomial tail certificate (Theorem H.2). In this appendix we write T for the threshold Γ, $\rho(c)$ for $\|c\|_{\mathsf{mul},2}$, and $\mathcal{C}_T^{\mathsf{fp}}$ for the accepted family $\mathcal{C}_\Gamma^{\mathsf{acc}}$.

Comparison with PikkuFold. PikkuFold [86, Section 5] studies the same operator-norm rejection idea, following the sketch in LaBRADOR [18]. Its Lemma 18 constructs bounded fixed-weight challenge families whose pairwise differences are invertible under the stated LS18 hypotheses. Its Lemma 19 proves that rejection preserves this property, gives a uniform distribution on the accepted set, and trades acceptance probability p against an expected trial count of $1/p$ and a loss of $-\log_2 p$ bits of cardinality. The concrete thresholds and acceptance rates in its Table 3 are calibrated and validated empirically. We add two certificates for specified challenge families: an exact arithmetic test that never accepts a challenge above the operator-norm threshold, and a proved lower bound on the number of challenges that pass it. Our mixed-magnitude base family fixes the number of coefficients of each magnitude, so it is a fixed-composition subset of PikkuFold's bounded fixed-weight family.

H.1 A Deterministic Fixed-Point Predicate

Let $d \geq 4$ be a power of two and set $R = \mathbb{Z}[X]/(X^d + 1)$. Let M_c denote negacyclic multiplication by $c \in R$. Evaluation at the odd $2d$-th roots of unity diagonalizes M_c, so

$$\rho(c) := \|M_c\|_{2\to 2} = \max_{0\leq k<d} |c(\zeta_k)|, \qquad \zeta_k = e^{(2k+1)\pi i/d}. \tag{287}$$

Since c has real coefficients, conjugate frequencies have equal magnitudes, so it suffices to check one frequency from each of the $d/2$ pairs.

Fix an integer precision $\nu \geq 0$ and a scale $\Delta = 2^\nu$. Choose signed integer tables $C[t]$ and $S[t]$ for $0 \leq t < 2d$, with a common nonnegative integer error bound ε, satisfying

$$|C[t] - \Delta\cos(\pi t/d)| \leq \varepsilon, \qquad |S[t] - \Delta\sin(\pi t/d)| \leq \varepsilon.$$

The bounds are obtained, and can be verified, in exact rational and integer arithmetic: Machin's identity gives a rational enclosure of π, octant reduction moves every angle to $[0, \pi/4]$, alternating Taylor series enclose sine and cosine, and the final conversion to scale Δ rounds outward.

For each frequency, define the exact accumulators

$$R_k := \sum_j c_j C[(2k+1)j \bmod 2d], \qquad I_k := \sum_j c_j S[(2k+1)j \bmod 2d].$$

If $L = \|c\|_1$ and $r = L\varepsilon$, the coordinate errors are at most r. It follows that

$$\Delta^2 |c(\zeta_k)|^2 \le U_k := R_k^2 + I_k^2 + 2(|R_k| + |I_k|)r + 2r^2. \tag{288}$$

For a positive rational threshold T, the fixed-point predicate accepts exactly when $U_k \le T^2\Delta^2$ for one frequency from each conjugate pair. Clearing the denominator of T makes this an integer comparison. $\mathcal{C}_T^{\mathsf{fp}}$ is the resulting subset of the base family $\mathcal{C}_0$.

The same enclosure also identifies a true-norm subset that is certainly accepted. Choose a nonnegative integer h with

$$h^2 > 8r^2, \qquad h < \Delta T, \qquad \tau_T := T - \frac{h}{\Delta}. \tag{289}$$

For a fixed base family, L and hence r are constant; the second condition ensures $\tau_T > 0$.

Theorem H.1 (Fixed-point sandwich). *The fixed-point predicate satisfies*

$$\{c \in \mathcal{C}_0 : \rho(c) < \tau_T\} \subseteq \mathcal{C}_T^{\mathsf{fp}} \subseteq \{c \in \mathcal{C}_0 : \rho(c) \le T\}. \tag{290}$$

Proof. The right inclusion follows directly from (288). For the left inclusion, let $v_k = \Delta(\operatorname{Re} c(\zeta_k), \operatorname{Im} c(\zeta_k))$. Each coordinate of (R_k, I_k) differs from v_k by at most r. Since $U_k = (|R_k|+r)^2 + (|I_k|+r)^2$, two applications of the triangle inequality give

$$\sqrt{U_k} \le \|v_k\|_2 + 2\sqrt{2}\, r < \Delta\tau_T + h = \Delta T.$$

Thus every challenge in the left-hand set passes at every frequency.

The first inclusion is used only to lower-bound the accepted cardinality. The second is the security property used when Akita replaces the general bound $\rho(c) \le \|c\|_1$ by the smaller threshold T.

H.2 An Exact Accepted-Support Certificate

Let a, b be nonnegative integers with $1 \le a + b \le d$, and let the base family $\mathcal{C}_0$ consist of the challenges with exactly a coefficients of magnitude one and b of magnitude two, with arbitrary signs. With $w = a + b$,

$$|\mathcal{C}_0| = \binom{d}{w}\binom{w}{b} 2^w. \tag{291}$$

Draw c uniformly from $\mathcal{C}_0$ and put $X_k := |c(\zeta_k)|^2$. The $d/2$ representative values X_k have the same marginal distribution, although they need not be independent: for power-of-two d, every odd integer is invertible modulo $2d$, so changing the odd frequency acts on $\mathcal{C}_0$ by a signed permutation of coefficient positions. Write $X := X_0$. Then $0 \le X \le B := \|c\|_1^2$, which is constant on $\mathcal{C}_0$.

Exact moments. The certificate records $M_m := \mathbb{E}[X^m]$ for $0 \le m \le 30$. These moments follow from an exact coefficient identity. With $\alpha = e^{\pi i/d}$ and $N_{\mathsf{supp}} = d!/(a!b!(d-a-b)!)$, sign averaging and support extraction give

$$M_m = \frac{(m!)^2}{N_{\mathsf{supp}}}[s^m t^m u^a v^b] \prod_{j=0}^{d-1} \left(1 + u\cosh(s\alpha^j + t\alpha^{-j}) + v\cosh(2s\alpha^j + 2t\alpha^{-j})\right). \tag{292}$$

The unnormalized moment $N_m := N_{\mathsf{supp}} M_m$ is a nonnegative integer. Indeed, sign averaging leaves only even powers of each signed coefficient, so N_m is an algebraic integer; invariance under the signed permutations above makes it fixed by every cyclotomic automorphism, hence rational, and a rational algebraic integer is an integer. The bound on X gives $0 \le N_m \le N_{\mathsf{supp}} B^m$. We compute N_m modulo primes $p > \max(d, 60)$ satisfying $p \equiv 1 \pmod{2d}$, representing α by an exact primitive $2d$-th root of unity. The Chinese remainder theorem recovers every N_m uniquely once the product of reconstruction primes exceeds $N_{\mathsf{supp}} B^{30}$. Dividing by N_{supp} yields the exact rational moments. An additional prime supplies an independent residue check.

Table 12: Certified operator-norm challenge filters.

d	counts (a,b)	T	h	$q_1^\star$	p_0	$\log_2(\lvert\mathcal{C}_0\rvert\, p_0)$
64	$(31,11)$	18	600	0.02390904	0.23491065	128.062439
128	$(31,0)$	13	351	0.00812072	0.48027391	128.563317

A rational tail majorant. Let Q be a rational polynomial of degree at most 30 satisfying

$$Q(x) \geq 0 \quad \text{on } [0,B], \qquad Q(x) \geq 1 \quad \text{on } [\tau_T^2, B]. \tag{293}$$

Then $\mathbf{1}_{\{X \geq \tau_T^2\}} \leq Q(X)$ pointwise, so for $Q(x) = \sum_i q_i x^i$ the marginal tail is bounded by the exact rational number

$$\Pr[X \geq \tau_T^2] \leq q_1^\star := \mathbb{E}[Q(X)] = \sum_i q_i M_i. \tag{294}$$

We search numerically for a polynomial and round it to rational coefficients; any rational Q satisfying (293) gives a valid certificate, so numerical optimality plays no role in the proof.

The two inequalities in (293) are checked over the full real intervals by exact polynomial arithmetic. One checker subdivides the intervals and proves that every Bernstein coefficient is nonnegative. The other uses a rational Sturm sequence to prove that the relevant polynomial (Q or $Q-1$) has no root in the interval, and checks a positive endpoint. These tests are sufficient, not necessary: a valid Q whose relevant polynomial has a zero in the interval fails the Sturm test.

Theorem H.2 (Accepted-support floor). *For a polynomial Q satisfying (293),*

$$\Pr[\rho(c) < \tau_T] \geq p_0 := 1 - \frac{d}{2} q_1^\star. \tag{295}$$

Consequently, $\lvert\mathcal{C}_T^{\mathsf{fp}}\rvert \geq \lvert\mathcal{C}_0\rvert\, p_0$. When $p_0 > 0$, independent uniform sampling from $\mathcal{C}_0$ until the fixed-point predicate accepts takes at most $1/p_0$ trials in expectation and returns a uniform element of $\mathcal{C}_T^{\mathsf{fp}}$.

Proof. The complement of $\{\rho(c) < \tau_T\}$ is the union, over the $d/2$ representative frequencies, of the events $\{X_k \geq \tau_T^2\}$. Each X_k has the law of X, so each event has probability at most $q_1^\star$ by (294), and a union bound proves (295). The cardinality claim follows from the left inclusion in (290) and uniformity of the base family. The trial count is geometric with success probability at least p_0, and all accepted elements have the same probability at each draw.

H.3 Concrete Certificates

For both dimensions we take $\nu = 48$ and the conservative error bound $\varepsilon = 4$, certified by the table construction above. Table 12 summarizes the resulting certificates. Its decimals are rounded displays of exact rational probabilities and their logarithms; the support claims are checked by the exact inequality $\lvert\mathcal{C}_0\rvert\, p_0 \geq 2^{128}$.

For $d = 64$, the base family has $\|c\|_1 = 53$, so $r = 212$ and $\tau_T = 18 - 600/2^{48}$. The degree-30 polynomial is certified by exact Bernstein positivity on 80 rational subdivisions of $[0, 2809]$ and $[\tau_T^2, 2809]$. For $d = 128$, the base family has $\|c\|_1 = 31$, so $r = 124$ and $\tau_T = 13 - 351/2^{48}$. Exact Sturm sequences certify positivity on $[0, 961]$ and $[\tau_T^2, 961]$; an independent 4096-subinterval Bernstein replay gives the same result.

The ancillary certificate data record the exact moment vectors, rational polynomial coefficients, and modular residues used in these calculations [66]. These certificates bound the cardinality of a fixed, witness-independent challenge family. Pairwise differences remain invertible because the filter only removes elements from the base family. The cardinality certificates do not by themselves establish a 128-bit schedule: coordinate-wise extraction losses, Fiat–Shamir queries, response-dependent nonce searches, and the remaining protocol errors are charged separately in Sections 10 and 12.

I Akita in Jolt

Jolt's PIOP produces evaluation claims about logical columns: instruction and memory addresses, increment digits, advice, and committed program data. Jolt commits to these columns through a few commitment objects. Columns of the execution trace share one prefix-packed polynomial; advice and committed-program objects keep their own commitments. At the final opening stage, Akita opens all these objects together, at their respective points, in one heterogeneous batched proof. This appendix describes that interface, including multi-point advice openings and joint proof framing [11], and its catalog-based setup-offloading extension. Appendix N records the exact Jolt and Akita revisions used in the measurements.

I.1 Committed Polynomials

The packed execution trace. Let $T = 2^t$ be the padded execution length and let $K = 2^a$ be the one-hot chunk alphabet. The logical trace columns have a common domain $\{0,1\}^{t+a}$, with coordinates ordered as cycle followed by address. They include the instruction-address columns, balanced increment digits, signed carry, bytecode-address columns, and RAM-address columns. The signed carry uses the same sparse layout, with its sign retained in the nonzero entry. We assign these columns to the first m slots of a fixed power-of-two capacity $C = 2^s$. Writing W_i for the restriction of the committed multilinear polynomial to slot i, prefix packing gives

$$W_{\mathsf{trace}}(\mathbf{z}, \mathbf{x}) = \sum_{i=0}^{C-1} \widetilde{\mathsf{eq}}(\mathbf{z}, \mathsf{bin}_s(i))\, W_i(\mathbf{x}). \tag{296}$$

Thus one commitment binds all trace columns, and the polynomial has $s + t + a$ variables.

The current layout uses $C = 64$ for $K = 16$ and $C = 32$ for $K = 256$. The corresponding Boolean evaluation vectors have 2^{t+10} and 2^{t+13} entries. These are the commitment-domain sizes: the prover constructs the commitment from sparse trace data without materializing the zero entries. The layout fixes the column order, dimensions, and variable permutation, and binds them through a digest checked against the commitment and verifier setup.

The honest encoder fills unused slots with zero. The application, however, uses only the restrictions $W_0, \ldots, W_{m-1}$ to the used slots, so the opening reduction below does not require an extracted polynomial to vanish on the unused slots.

Advice and committed programs. Advice is represented by a separate bounded-dense polynomial of 64-bit words. Trusted advice is committed during preprocessing. Untrusted advice is committed at the start of proving, before the trace commitment, and joins the final opening under its own commitment rather than being copied into the trace polynomial. In committed-program mode, each bytecode chunk and the initial program image likewise form separate bounded-dense commitment objects. A small object can occupy one used prefix slot in a larger commitment domain to meet the backend's supported commitment geometry.

Each object has its own logical domain, layout digest, and commit-time parameters, fixed before its opening point is known, as in Section 4.6. Committing untrusted advice before the trace does not mean that it was known during preprocessing.

I.2 From Column Claims to Commitment Openings

Jolt's preceding claim reductions arrange that the columns of each committed polynomial are opened at one common logical point. For the trace, the adapter moves the cycle coordinates before the address coordinates and checks that the resulting leaf points agree. Different objects can retain different points and arities.

Fix one object W with m used slots in capacity $C = 2^s$. Suppose the logical claims are $W_i(\mathbf{r}) = y_i$ for $0 \le i < m$. After binding the layout, $\mathbf{r}$, and all y_i to the transcript, the verifier samples a selector $\boldsymbol{\zeta} \in \mathbb{F}^s$. It computes the claim on the packed polynomial

$$W(\boldsymbol{\zeta}, \mathbf{r}) = y_{\mathsf{pack}}, \qquad y_{\mathsf{pack}} := \sum_{i=0}^{m-1} \widetilde{\mathsf{eq}}(\boldsymbol{\zeta}, \mathsf{bin}_s(i))\, y_i. \tag{297}$$

This step evaluates public equality weights and introduces no additional product sum-check. For a single-slot capacity, it is simply the original opening.

Lemma I.1 (Prefix reduction without a padding predicate). *Fix a packed multilinear polynomial W, a logical point $\mathbf{r}$, and claims $y_0, \ldots, y_{m-1}$ before sampling a uniform selector $\boldsymbol{\zeta} \in \mathbb{F}^s$. If $W_i(\mathbf{r}) \neq y_i$ for some used slot $i < m$, then (297) holds with probability at most $s/|\mathbb{F}|$. No restriction on the unused slots of W is required.*

Proof. The difference between the two sides, as a polynomial in the selector, is

$$D(\mathbf{z}) := W(\mathbf{z}, \mathbf{r}) - \sum_{i=0}^{m-1} \widetilde{\mathsf{eq}}(\mathbf{z}, \mathsf{bin}_s(i))\, y_i.$$

It is multilinear in s variables. At the Boolean selector for a false claim with $i < m$, $D(\mathsf{bin}_s(i)) = W_i(\mathbf{r}) - y_i \neq 0$, so D is nonzero regardless of the unused slots. The polynomial identity bound gives the result.

Consequently, once the PCS binds the packed multilinear polynomial, prefix reduction binds the claimed evaluations on its used slots with the error in Lemma I.1. For several objects, these errors add over their selector draws; their Fiat–Shamir accounting is separate from the backend PCS error. Booleanity, one-hot validity, and the relations connecting decoded values to the execution remain obligations of the Jolt PIOP.

I.3 One Heterogeneous Akita Proof

Each prefix reduction yields one PCS opening statement consisting of an original commitment, a point, and a claimed value. The final batch lists the present objects in the canonical order

$$[\, U,\ A,\ B_0, \ldots, B_{c-1},\ P,\ W_{\mathsf{trace}} \,],$$

where U and A are untrusted and trusted advice, the B_j are bytecode chunks, and P is the initial program image. Objects absent from the configured execution are omitted. The trace is the final group, and all earlier groups keep their commit-time descriptors. Akita's multi-group opening protocol authenticates the whole list in one proof, using the group-local points and shapes of Section 5. No commitment homomorphism or common point across objects is required.

The verifier checks that every required object and logical claim is present, that the object roles and order are canonical, and that the commitment metadata match the public layouts. Advice and program claims must refer to their original commitments. The verifier binds the ordered commitments, points, values, and schedule selection before deriving the backend transcript. The same construction handles an execution with only the trace group.

The Jolt wrapper stores a fixed 32-byte schedule selection and Akita's canonical headerless proof body. The verifier resolves the selection against its admitted schedule before deriving the backend decoding bounds. Proof-size reporting distinguishes that backend body, the selection-plus-body payload, and the complete serialized Jolt proof, including its enclosing framing and PIOP messages; a size reported for the batched PCS proof is not the size of the entire Jolt proof.

I.4 Schedules and Setup Offloading

The trace size and the shapes of separately committed objects determine the grouped opening profile. Jolt supplies a schedule for that exact profile, with commit-time parameters preserved for every earlier object. The schedule determines the recursive witness, opening representations, response bounds, setup requirements, and terminal check.

Catalog-based integration. The setup-offloading extension stores schedules in external catalogs maintained by the application, and adapts scalar trace schedules to the grouped profiles required by Jolt. During preprocessing, the application admits the base catalog and the exact rows needed for advice and committed-program objects. The resulting immutable catalog is stored with the verifier setup, validated again when a transported setup is restored, and bound into the setup transcript by its digest. The proof selects a row from this catalog rather than supplying new schedule parameters.

Adapting a scalar trace row keeps its recursive depth, dimensions, block geometry, opening parameters, relation modes, and choice of direct or offloaded folds. It rederives the grouped root's shared opening matrix and every dependent witness length, rank, response bound, setup-prefix length, and proof cost, and the adapted row must pass the ordinary schedule and catalog admission checks. If the kept structure cannot accommodate the new objects, preprocessing fails. The adapted row need not be as good as the result of a fresh unconstrained search.

For an offloaded fold, the setup-product check of Section 9 leaves an opening of the authenticated public setup prefix. The next fold checks that opening alongside the recursive witness. The selected schedule determines how many such steps occur; the final setup opening must be checked before the terminal. This uses the same multi-group interface as the application batch.

Where offloading becomes worthwhile is a policy choice made when the catalogs are built; proving and verification follow the selected row. In the measured M4 Max configuration of Section 13.4, setup offloading first appears at padded trace length $T = 2^{21}$. This crossover belongs to that configuration and its pinned revisions; it is not a universal threshold.

I.5 Transcript Grinding in the Backend

The Akita dependency used by the merged Jolt integration already implements two distinct nonce mechanisms.[13] Fold-response search finds a challenge whose honest response satisfies the scheduled representation and norm bounds. Transcript proof-of-work instead precedes selected algebraic challenge queries: the verifier checks a zero-bit predicate on one transcript output before drawing the protocol challenge from a separate output.

Section 10.1 gives the classical online random-oracle work bound for this mechanism. It is not a QROM theorem or a complete end-to-end security estimate, which must also account for the remaining Jolt and Akita errors and computational reductions.

A schedule-derived plan fixes the query order and nonce widths, and its digest is bound into the instance descriptor. Both nonce mechanisms draw from one packed nonce stream per proof, and the verifier must read exactly the entries that the plan specifies. Sparse fold coordinates are derived by separately indexed oracle queries from a fixed group root, so changing one coordinate leaves the others fixed as required by coordinate-wise extraction. The outer Jolt PIOP, its prefix-selector draws, and the backend proof remain distinct parts of the composed transcript.

J Corrections and Security Mismatches in Prior Work

The security of a lattice-based proof system depends on the exact algebraic relations proved by the protocol and on the concrete maps and parameters instantiated by its implementation. This appendix records two issues in public protocol descriptions and several mismatches between released implementations and their stated security arguments. We state each issue at the narrowest level supported by the evidence; none of the implementation findings below is presented as an end-to-end attack on the complete proof system.

J.1 Corrections to Protocol Descriptions

The Base-Field Optimization in Hachi Hachi first gives a generic transformation from evaluation claims over an extension field to evaluation claims over a cyclotomic ring [80, Section 3.1]. It then proposes a shorter transformation for the common case in which the committed polynomial has coefficients in the base field but is evaluated at an extension-field point [80, Section 3.2]. The specialization reduces the number of variables in the resulting ring-valued polynomial, but the binding check stated in the public paper does not establish that the claimed partial evaluations are correct.

Let $K := \mathbb{F}_q$, let E/K have degree $k = 2^\kappa$, and split an ℓ-variable polynomial f into the base-field polynomials $f_y(\mathbf{W}) := f(y, \mathbf{W})$ for $y \in \{0,1\}^\kappa$. At an opening point $(\mathbf{r}_{\mathsf{head}}, \mathbf{r}_{\mathsf{tail}})$, write

$$T_y := f_y(\mathbf{r}_{\mathsf{tail}}) \in E$$

[13] See Akita PR #460, merged as `4505404b5`, and its transcript-grinding specification.

for the honest partial evaluations. The prover supplies claimed partials $S_y \in E$, and the verifier checks their recombination against the original opening value:

$$v = \sum_{y \in \{0,1\}^\kappa} \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{head}}, y) S_y. \tag{298}$$

The optimization also fixes a K-basis $(\beta_y)_{y \in \{0,1\}^\kappa}$ of E, forms the packed polynomial

$$g(\mathbf{W}) := \sum_y \beta_y f_y(\mathbf{W}),$$

and reduces its claimed evaluation to the ring using Hachi's generic transformation. At the level of the partials, this authenticates only the second combination

$$g(\mathbf{r}_{\mathsf{tail}}) = \sum_y \beta_y S_y. \tag{299}$$

These two checks do not bind the individual S_y. Set $\Delta_y := S_y - T_y$ and $\chi_y := \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{head}}, y)$. A false set of partials passes both displayed checks whenever

$$\sum_y \chi_y \Delta_y = 0, \qquad \sum_y \beta_y \Delta_y = 0. \tag{300}$$

These are two E-linear equations in k extension-valued discrepancies. For $k > 2$, including the degree-4 setting used in the concrete Hachi and Akita parameters, the corresponding map from E^k to E^2 has a nontrivial kernel. The K-linear independence of the basis elements β_y does not rule out such a kernel because the discrepancies Δ_y lie in E.

This joint-kernel argument establishes nonuniqueness of the partials while preserving the original claimed value. A false value requires a different discrepancy: one in the kernel of $\boldsymbol{\beta}$ but outside the kernel of $\boldsymbol{\chi}$. Such a discrepancy exists whenever these two E-linear functionals are not proportional. Explicitly, choose a, b with $\chi_a\beta_b - \chi_b\beta_a \neq 0$ and set

$$\Delta_a = \beta_b, \qquad \Delta_b = -\beta_a, \qquad \Delta_y = 0 \quad (y \notin \{a, b\}).$$

Then the packed claim is unchanged, while the first displayed check accepts the altered value $v' = v + \chi_a\beta_b - \chi_b\beta_a \neq v$. This identifies what fails in the displayed base-to-extension bridge; it does not assert acceptance by every other check of the complete protocol.

We use the tensor-algebra ring-switching reduction of Diamond and Posen [39, Theorem 3.5] to authenticate these partials. The reduction expands the claimed extension-valued columns into their base-field coordinates, changes from the column view to the row view in the tensor algebra, and batches the resulting row identities with a fresh verifier challenge. Basis independence is then applied only to base-field coordinates, which gives a sound realization of the base-field specialization.

The currently public Hachi ePrint still contains the check above in Section 3.2. The Hachi authors have informed us that they are aware of the issue and have corrected subsequent manuscript and implementation versions, which were not yet publicly disseminated at the time of writing [81]. The issue is confined to the Section 3.2 specialization and does not apply to Hachi's generic extension-field transformation in Section 3.1.

We do not include Hachi as an evaluation baseline. Its public artifact reports prover and verifier measurements for the first folding round rather than an end-to-end recursive opening [80, Section 5], and our evaluation is restricted to artifacts that can execute the complete PCS workflow.

Challenge Ordering in the First Grand Danois Version The first public version of Grand Danois sampled a row-combination challenge before committing to the recursive witness [58]. A prover could therefore choose a nonzero residual in the known projection kernel; the usual random-linear-combination bound requires the residual to be fixed first. The current version removed this reduction: it commits to the witness before sampling the field challenges and checks a power-row projection of the coefficient relation [57]. This is the one-product functional identified in (261). The authors credit Quang Dao for identifying the earlier issue. We retain this account of the archived version; our comparisons use the repaired construction.

J.2 Released Implementations

While preparing our evaluation, we audited the released implementations most relevant to Akita and found several concrete mismatches between the implemented maps or parameters and the arguments given in the corresponding papers. We distinguish these findings from subsequent repairs and from ordinary differences in concrete-security conventions: prior systems use different lattice-reduction cost models, target security levels, and methods for allocating a security budget across protocol components. Those choices affect how we report and normalize our evaluation, but they are not by themselves security defects; we discuss them separately in Appendix G and Section 13. Our RoKoKo measurements use the corrected implementation described below.

Greyhound and RoKoKo are direct baselines for our evaluation. We also include Orthus, although it targets batch verification rather than polynomial commitments, because it uses a commitment-compression technique similar to Akita's. Its first map compresses a wide source into an intermediate commitment, and its second map compresses a binary decomposition of that commitment into a short public payload. Orthus therefore provides a direct comparison for our compression maps, whose security estimates include the full input width of each stage.

Truncated-Output Ring-SIS in LaBRADOR and Greyhound The LaBRADOR commitment and the Greyhound polynomial commitment scheme are analyzed using uniformly random matrices over a base ring, with binding reduced to Module-SIS [18,82]. Their implementations use a more structured commitment map. Instead of sampling and applying each Module-SIS row independently, they pack several base-ring elements into an extension ring and compute the corresponding commitment by extension-ring multiplication.

Concretely, let

$$S = \mathbb{Z}_q[Y]/(Y^n + 1)$$

be the base ring, let κ be the desired commitment rank, and let p be the smallest power of two satisfying $p \geq \kappa$. The implementation works over the extension

$$T = S[Z]/(Z^p - Y).$$

Writing the packed key and message blocks as $a_b(Z), z_b(Z) \in T$, the commitment returned by the implementation has the form

$$\mathsf{Com}(z) = \pi_{<\kappa}\left(\sum_b a_b(Z) z_b(Z)\right),$$

where $\pi_{<\kappa}$ retains only the first κ coordinates of the extension-ring product.

When $\kappa = p$, this is a complete product in T. Its binding can be stated as an ordinary rank-one Ring-SIS problem over the larger ring T. This is already different from the base-ring Module-SIS instance used in the papers, but it remains a standard SIS assumption.

The more delicate case is $\kappa < p$. A collision then establishes only

$$\pi_{<\kappa}\left(\sum_b a_b(Z) \Delta z_b(Z)\right) = 0.$$

The remaining $p - \kappa$ coordinates need not vanish. Thus, the collision is not a Ring-SIS solution in T. Nor is it a solution for a uniformly random rank-κ Module-SIS matrix over S, since the retained rows are correlated rotations of the same packed key material. We refer to the resulting problem as *truncated-output Ring-SIS*.

Greyhound records the first half of this optimization explicitly. Its implementation section states that commitments are computed over extension rings, viewed as vector spaces over the base ring, in order to reduce the randomness sampled for the commitment matrices [82, Section 6.1, p. 30]. It does not state that the extension degree is rounded up to the next power of two, that only the requested κ coordinates are returned, or that the returned coordinates are therefore a truncation whenever $\kappa < p$. Meanwhile, the security proof embeds a uniformly random Module-SIS challenge. The missing step is a reduction covering the truncated output map. We do not know an attack exploiting this structure, but the published Module-SIS theorem and the corresponding Module-SIS estimates do not establish binding for the implemented map.

This distinction affects our evaluation directly. A standard-assumption comparison should either use the uniform Module-SIS commitment analyzed in the paper or retain the complete extension-ring output and estimate the resulting Ring-SIS instance. The released truncated implementation can still be measured, but only under an explicit truncated-output Ring-SIS assumption.

RoKoKo: Radix Recomposition and Upstream Correction We identified a radix-recomposition error in an earlier RoKoKo implementation. For ℓ digits in radix 2^k, recomposition has Euclidean operator norm

$$\rho_{k,\ell} = \left(\sum_{i=0}^{\ell-1} 2^{2ki} \right)^{1/2}.$$

The earlier implementation used $k^{\ell-1}$ instead. This underestimated extraction norms in the lattice-security estimates and in checks used to justify lifting modular norm identities to integer bounds. The original calculations and affected parameters are recorded in issue 94.[14]

The RoKoKo authors repaired the implementation, including its recomposition bounds, commitment packing, and projection-norm accounting, in revision `ba83b317`. Our measurements use revision `5caba472`, which includes these corrections. The September 9, 2026 revision of the RoKoKo paper also updates the exposition and experiments and acknowledges the initial implementation bugs [62]. The earlier failed checks therefore do not describe the implementation benchmarked here.

Remaining security distinctions. The corrected implementation retains RoKoKo's native security assumptions and parameter choices. Its structured vSIS assumption is heuristically estimated through generic SIS attacks, and its estimator uses GSA with classical MATZOV accounting. Akita instead uses standard Module-SIS with LGSA and quantum ADPS16 accounting; Appendix G explains why the resulting bit estimates are not interchangeable.

RoKoKo uses a roughly 50-bit prime q and quadratic-extension challenge slots of size $q^2 \approx 2^{100}$. Its stated statistical soundness target is approximately 2^{-100} [62, Section 9.3]. We refer to this as its *100-bit statistical soundness target*, separately from its 128-bit lattice-security target. Its approximate low-norm challenge sampling has a further qualification: the paper estimates noninvertibility at about 2^{-94}, using a heuristic estimate and citing Cyclo for further rigorous analysis [62, Section 9.1]. This estimate concerns a bad event in challenge sampling and does not by itself establish a forgery attack of that cost. These current distinctions, rather than the repaired radix error, are why our native-parameter comparison is not security matched.

Orthus: Outer-Commitment Width and Upstream Correction The outer-commitment sizing issue below affected an earlier Orthus implementation and has been fixed. Commit `5fc08b5a` increased the middle-commitment output from 32 to 64 coefficients and the projection-commitment output from 32 to 256. The authors confirmed the repair and announced a corresponding paper update on 17 September 2026.[15] With the default 38-bit modulus, the corrected compressed middle commitments contain $64 \cdot 38/8 = 304$ bytes each, compared with Akita's 128 bytes: 2.375 times as large. The separate projection commitment contains $256 \cdot 38/8 = 1{,}216$ bytes. These sizes count the packed commitment coefficients. We retain the original width calculation to explain the issue; the estimates below concern the pre-fix parameters and do not apply to the corrected implementation.

Orthus begins with a protocol whose prover messages are too large to send in the clear. Its communication compression has two layers. First, the prover digit-decomposes these messages and commits to them with Ajtai commitments; Orthus calls the resulting images *middle commitments*. The final LaBRADOR statement proves that these commitments open to the protocol messages and that the messages satisfy the Orthus verifier's relations. Second, rather than send the middle commitments themselves, the prover decomposes their coefficients into bits and commits once more, producing the *outer commitments* that appear in the proof [20, Section 5.1]. The random projection messages take a shorter route: they are bit-decomposed and committed directly by the same outer map, without an intervening middle commitment.

[14] https://github.com/lattice-arguments/rokoko/issues/94.
[15] https://github.com/lazer-crypto/labrador/issues/1#issuecomment-5718109866.

Outer input	M over R_q	Width over S_q	Scalar shape	ADPS16	BDGL16
Middle commitment	114	912	$32 \times 29{,}184$	100.448	134.681
Projection message	8,576	68,608	$32 \times 2{,}195{,}456$	27.448	67.968

Table 13: Estimated base-2 attack costs for Orthus's pre-fix outer commitments. Both calculations use coefficient bound one, the LGSA basis-shape model, and no ignored coordinates. ADPS16 denotes the classical Core-SVP cost model [5]; BDGL16 denotes the full operation-count model [15].

This last binary decomposition is what makes aggressive compression seem possible. Two valid binary openings differ coefficientwise in $\{-1, 0, 1\}$, so their collision vector has coefficient ℓ_∞-norm one. The original Orthus analysis invoked the Dilithium SIS analysis, explicitly ignored the algebraic structure, and concluded that 32 output equations suffice [20, Section 5.1 and Appendix E]. The input width was absent from that calculation. Yet SIS security depends on the modulus, output rank, norm bound, and number of input columns. A fixed map to 32 field elements cannot securely compress binary vectors of arbitrary width merely because their collision bound is one.

In the original implementation, the outer commitment is computed over

$$R_q = \mathbb{Z}_q[X]/(X^{256} + 1),$$

but only the coefficients at positions $0, 8, 16, \ldots, 248$ are transmitted. Setting $Y = X^8$, these coefficients form an element of the subring

$$S_q = \mathbb{Z}_q[Y]/(Y^{32} + 1).$$

After splitting each degree-256 input polynomial into its eight residue lanes, the public commitment can be written as

$$\mathsf{Com}_{\mathsf{out}}(z) = \sum_{\ell=1}^{M} \sum_{j=0}^{7} b_{\ell,j}(Y) z_{\ell,j}(Y) \in S_q.$$

Thus, the outer commitment is exactly a rank-one Ring-SIS map over the degree-32 ring S_q, with ring width $8M$. No new structured assumption is needed for this endpoint, but the full width $8M$ must be included in the security estimate.

The pre-fix Falcon aggregation benchmark used 2^{13} signatures, $q = 2^{38} - 107$, and ring degree 256. Each ordinary middle commitment has module rank $t_2 = 3$. Writing each of its coefficients in 38 bits therefore gives

$$M_{\mathsf{mid}} = 38 t_2 = 114$$

binary elements of R_q. After the lane decomposition above, this is a degree-32 rank-one Ring-SIS instance of width $8M_{\mathsf{mid}} = 912$, or equivalently a scalar SIS matrix with 32 rows and 29,184 columns. The directly committed projection message is much wider: it contains 8,576 binary elements of R_q, giving degree-32 ring width 68,608 and scalar width 2,195,456.

The ordinary middle-to-outer commitment already falls below 128 bits under the ADPS16 classical Core-SVP convention used for Akita's comparison tables. The projection-to-outer commitment falls below 128 bits under both models, reaching only 27.448 ADPS16 bits and 67.968 BDGL16 bits. These estimates expand the exact degree-32 map as a generic scalar SIS instance; they do not claim any additional attack from its ring structure. They show that even this generic model does not support the claimed 128-bit binding level at the original implemented width.[16]

The issue is therefore not the use of a small terminal commitment by itself. The wide source must first be compressed by a map sized for that source. Only the bounded intermediate image can then be compressed into a small terminal payload. Akita follows this staged approach and validates the input width of each compression map separately.

[16] See https://github.com/lazer-crypto/labrador/issues/1 for the complete derivation and estimator reproduction.

PikkuFold Parameter Updates We reported a missing challenge operator-norm factor in the backward-extraction recurrence used by an earlier PikkuFold parameter notebook. The authors fixed the issue by revising the norm-growth analysis and parameters in pull requests 3 and 4.[17]

K Response Bounds and Parameter Estimates

Response bounds have two roles in Akita. The verifier enforces them to bound extracted witnesses, while the honest prover must satisfy them without too many retries. This appendix supplies the estimates for the second role, using the admission predicates and history-conditional completeness framework of Section 10.2. We first prove coefficient and squared-norm bounds for the actual response distributions and derive completeness from them. We then give the response model used to propose tighter parameters and estimate retry cost; its guarantees for integer responses remain conditional on the stated model premise. Finally, we give a pointwise alternative for propagating digit squared norms and explain how the response digit depth is chosen. Changing a bound by any of these methods requires validating the complete resulting schedule, since the next witness and its commitment parameters can change with it.

Rademacher sums and Hoeffding bounds. In several bounds below, once all non-sign data are fixed, the remaining randomness is a collection of independent fair signs. The triangle inequality bounds such a signed sum as if all contributions were aligned, which is far too pessimistic; Hoeffding's inequality quantifies the cancellation.

Definition K.1 (Rademacher variables). *A* Rademacher random variable *is a random variable ε with*

$$\Pr[\varepsilon = 1] = \Pr[\varepsilon = -1] = \tfrac{1}{2}.$$

Let J be a finite index set, let $(\varepsilon_j)_{j\in J}$ be independent Rademacher random variables, and let $(w_j)_{j\in J}$ be deterministic real weights. The random variable

$$X = \sum_{j\in J} \varepsilon_j w_j$$

is a Rademacher sum. *Its variance proxy is*

$$V_X := \sum_{j\in J} w_j^2.$$

The weights are often deterministic only after conditioning on public or previously sampled data. If, after such conditioning, the remaining random variables are independent Rademachers with fixed coefficients, we call the conditioned quantity a Rademacher sum. A tail bound proved for every conditioning event then also holds unconditionally, by averaging.

Lemma K.2 (Hoeffding bound for Rademacher sums). *Let*

$$X = \sum_{j\in J} \varepsilon_j w_j$$

be a Rademacher sum with variance proxy $V_X = \sum_{j\in J} w_j^2$. Then for every $t \geq 0$,

$$\Pr[|X| > t] \leq 2\exp\left(-\frac{t^2}{2V_X}\right),$$

with the convention that the right-hand side is 0 when $V_X = 0$.

Lemma K.2 replaces the linear worst-case scale $\sum_j |w_j|$ by the square-root scale $\sqrt{V_X}$, up to the logarithmic factor set by the target failure probability. The fold sizing of Section 5.5 uses it to size digit intervals from squared weights.

[17] https://github.com/osdnk/pikku/issues/2; https://github.com/osdnk/pikku/pull/3; https://github.com/osdnk/pikku/pull/4.

K.1 Bounds for Honest Responses

Fix the honest transcript prefix before a response nonce is sampled, so that the source vectors are fixed. Our bounds concern the resulting integer convolutions. Coefficientwise centering modulo q can only decrease their absolute coefficients and squared norm.

Dense and One-Hot Sources without Filtering For dense sources, we condition on the challenge supports and magnitudes and use the remaining independent signs. A canonical one-hot source admits a sharper bound because only a few source coefficients can contribute to one output position.

Lemma K.3 (Dense-source response bound). *Fix an honest response*

$$\mathbf{z} = \sum_{a \in \mathcal{Q}} c_a * \mathbf{s}_a$$

and condition on the sources and on the support and absolute value of every nonzero challenge coefficient. Suppose the remaining challenge signs are independent Rademacher variables. For output coefficient u, write

$$V_u := \sum_{a \in \mathcal{Q}} \sum_{v \in \mathrm{supp}(c_a)} |c_{a,v}|^2 s^2_{a,u \ominus v}, \tag{301}$$

where $u \ominus v$ includes the deterministic sign of negacyclic wraparound. Then, for every $t > 0$,

$$\Pr[|z_u| > t] \le 2\exp(-t^2/(2V_u)). \tag{302}$$

If the response has N coefficients and $V_u \le V$ for all u, then

$$\Pr[\|\mathbf{z}\|_\infty > t] \le 2N\exp(-t^2/(2V)). \tag{303}$$

In particular, if $\|\mathbf{s}_a\|_\infty \le \sigma_\infty$ and $\|c_a\|_2^2 \le \kappa_2^2$ for every a, one may take $V = |\mathcal{Q}|\kappa_2^2\sigma_\infty^2$.

Proof. For fixed u, negacyclic convolution expresses z_u as a Rademacher sum whose squared weights sum to (301). Lemma K.2 gives (302); a union bound over the N coordinates gives (303). The uniform value of V follows by replacing every selected source coefficient by σ_∞ and summing the challenge squared norms.

Lemma K.4 (Fixed-weight one-hot moment bound). *Let $x \in \{0,1\}^d$ have Hamming weight at most ν. Choose disjoint uniform supports $S_1, S_2 \subseteq [d]$ of fixed sizes n_1, n_2, and choose independent Rademacher signs $(\varepsilon_i)_{i \in S_1 \cup S_2}$. For*

$$X := \sum_{i \in S_1} \varepsilon_i x_i + 2\sum_{i \in S_2} \varepsilon_i x_i,$$

every $\lambda > 0$ satisfies

$$\mathbb{E}[e^{\lambda X}] \le \left(1 + \frac{\nu}{d}(\cosh\lambda - 1)\right)^{n_1} \left(1 + \frac{\nu}{d}(\cosh(2\lambda) - 1)\right)^{n_2}. \tag{304}$$

Proof. Let $\nu_0 := \|x\|_0 \le \nu$. Generate the supports by an ordered sample of $n_1 + n_2$ distinct positions: the first n_1 form S_1, and the rest form S_2. Let Z_j indicate that sampled position j lies in the support of x. For any set J of r sample slots,

$$\mathbb{E}\left[\prod_{j \in J} Z_j\right] = \frac{(\nu_0)_r}{(d)_r} \le (\nu/d)^r,$$

where $(a)_r := a(a-1)\cdots(a-r+1)$ and $(a)_0 := 1$. Averaging the signs gives $\mathbb{E}\prod_j(1 + a_j Z_j)$, where $a_j = \cosh\lambda - 1$ on the first n_1 slots and $a_j = \cosh(2\lambda) - 1$ on the remaining slots. All a_j are nonnegative. Expanding this product and applying the displayed bound term by term gives $\prod_j(1 + (\nu/d)a_j)$, which is (304).

Theorem K.5 (Canonical one-hot response bound). *Fix a root source formed from unit one-hot chunks of length K, represented in rings of dimension d, where either $K \mid d$ or $d \mid K$, and let*

$$\nu_{\mathsf{oh}} := \max\{1, d/K\}$$

be the maximum number of canonical nonzero coefficients from one source class in a ring. Suppose $\nu_{\mathsf{oh}} \le d_c$ and the challenge is sampled in dimension d_c with n_1 coefficients in $\{\pm 1\}$ and n_2 coefficients in $\{\pm 2\}$, with each fixed-weight support sampled without replacement and the two supports disjoint. For $a \in \{1, 2\}$ and $\lambda > 0$, define

$$M_a(\lambda) := 1 + \frac{\nu_{\mathsf{oh}}}{d_c}\big(\cosh(a\lambda) - 1\big). \tag{305}$$

For coefficient u, let $\mathcal{Q}_u$ index the independently challenged source contributions to z_u, and put $W_u = |\mathcal{Q}_u|$ and $W = \max_u W_u$. These are challenge/source pairs indexed by group, claim, and block (g, c, i), not opening groups: a single group with B independently challenged blocks can contribute B pairs. Then

$$\Pr[|z_u| \ge t] \le 2 \inf_{\lambda > 0} \exp\big(W_u\big(n_1 \log M_1(\lambda) + n_2 \log M_2(\lambda)\big) - \lambda t\big)\,. \tag{306}$$

For a response with N coefficients, replacing W_u by W and multiplying by N bounds the probability that any coefficient exceeds t. The deterministic bound

$$\|\mathbf{z}\|_\infty \le W \min\{2\nu_{\mathsf{oh}}, n_1 + 2n_2\} \tag{307}$$

is valid simultaneously, so the schedule may take the smaller of the deterministic bound and any threshold certified by (306).

Proof. For one fixed output coefficient, the challenge support samples a population with at most ν_{oh} active unit entries. Lemma K.4 bounds its moment-generating function by $M_1(\lambda)^{n_1} M_2(\lambda)^{n_2}$. Multiplying over the W_u independently sampled challenge/source contributions and applying the two-sided Chernoff bound gives (306). The union bound gives the joint statement. Young's two coefficient inequalities give $\|c * s\|_\infty \le \min\{\|c\|_\infty \|s\|_1, \|c\|_1 \|s\|_\infty\}$, which yields (307).

Squared-Norm and Coefficient Bounds for Filtered Challenges A filter changes the challenge distribution, and an operator filter need not preserve the independence, centering, and fixed-weight structure used by Lemma K.3 and Theorem K.5. Unless that structure is proved for the accepted distribution, the schedule must analyze the accepted distribution directly, as the following bounds do, or treat the resulting joint admission probability as modeled. These bounds need independent accepted challenges and negation-closed accepted families, not independent coefficient signs within a challenge. We begin with the expected response squared norm.

Lemma K.6 (Conditional response squared norm). *For a distribution $\mathcal{D}$ over integer challenge polynomials, let $\mathsf{Mul}(c)$ be the real negacyclic multiplication matrix of c and define*

$$\nu_2(\mathcal{D}) := \lambda_{\max}\big(\mathbb{E}_{c \leftarrow \mathcal{D}}[\mathsf{Mul}(c)^{\mathsf{T}} \mathsf{Mul}(c)]\big)\,. \tag{308}$$

Then every fixed source vector $\mathbf{s}$ satisfies

$$\mathbb{E}_{c \leftarrow \mathcal{D}}[\|c * \mathbf{s}\|_2^2] \le \nu_2(\mathcal{D}) \|\mathbf{s}\|_2^2. \tag{309}$$

If the challenges c_a are independent and centered, then

$$\mathbb{E}\left[\left\|\sum_{a \in \mathcal{Q}} c_a * \mathbf{s}_a\right\|_2^2\right] \le \sum_{a \in \mathcal{Q}} \nu_2(\mathcal{D}_a) \|\mathbf{s}_a\|_2^2. \tag{310}$$

These statements also apply to an operator-filtered challenge family when $\mathcal{D}$ denotes its actual accepted distribution.

Proof. The first identity is $\mathbb{E}[\mathbf{s}^{\mathsf{T}}\mathsf{Mul}(c)^{\mathsf{T}}\mathsf{Mul}(c)\mathbf{s}]$; the largest eigenvalue bounds the quadratic form. Centering and independence make every cross term in the squared norm of the sum vanish, giving the second claim.

Lemma K.7 (Certified squared norm under a symmetric filter). *Let $\mathcal{C}_0$ be a fixed-weight signed sparse base family of challenge squared norm κ_2^2. Let $\mathcal{C}^{\mathsf{acc}} \subseteq \mathcal{C}_0$ be a nonempty accepted family closed under negation, containing a fraction at least $\underline{\alpha} > 0$ of $\mathcal{C}_0$, and with multiplication norm at most Γ. For its uniform distribution $\mathcal{D}$, one has*

$$\mathbb{E}[c] = 0, \qquad \nu_2(\mathcal{D}) \le \min\{\Gamma^2, \kappa_2^2/\underline{\alpha}\}.$$

Consequently, independent accepted challenges and deterministic source squared-norm envelopes $\|\mathbf{s}_a\|_{2,\mathrm{coef}}^2 \le E_a$ give

$$\mathbb{E}\left[\left\|\sum_a c_a * \mathbf{s}_a\right\|_{2,\mathrm{coef}}^2\right] \le \mu := \sum_a \min\{\Gamma_a^2, \kappa_{2,a}^2/\underline{\alpha}_a\} E_a.$$

For a nonnegative integer squared-norm bound S, the probability of exceeding it is at most $\mu/(S+1)$. The same upper bound applies after coefficientwise centering modulo q.

Proof. Negation pairs show that the mean is zero. Under the unfiltered base family, independent fair signs make the coefficient cross terms vanish, while the fixed total squared norm gives $\mathbb{E}[\mathsf{Mul}(c)^{\mathsf{T}}\mathsf{Mul}(c)] = \kappa_2^2 I$. Conditioning a nonnegative quadratic form on an event of probability at least $\underline{\alpha}$ increases its expectation by at most $1/\underline{\alpha}$. The pointwise operator bound gives the other bound. Apply Lemma K.6. Since squared coefficient norms are integers, failure of the bound means that the norm is at least $S+1$; Markov's inequality completes the proof.

The coefficientwise and squared-norm conditions are combined through Corollary K.10. For example, a deterministic coefficient envelope can certify encodability while this lemma certifies the additional Euclidean bound.

Lemma K.8 (Coefficient tails under symmetric filters). *Fix the sources in an honest response $\mathbf{z} = \sum_a c_a * \mathbf{s}_a$, first evaluated over the integers. Suppose that the c_a are independent, each uniform on a nonempty, negation-closed fixed subset of a signed sparse base family of squared norm $\kappa_{2,a}^2$ and challenge dimension d_a. Let $\underline{\alpha}_a > 0$ lower-bound its accepted fraction. For each output coefficient u, let $E_{a,u}^{\mathsf{loc}}$ bound the squared norm of the d_a source coefficients that can enter $(c_a * \mathbf{s}_a)_u$, and let $m_{a,u}$ bound its absolute value for every accepted challenge. Set*

$$V := \max_u \sum_a \min\left\{m_{a,u}^2, \frac{\kappa_{2,a}^2 E_{a,u}^{\mathsf{loc}}}{d_a \underline{\alpha}_a}\right\}, \qquad M := \max_{a,u} m_{a,u}.$$

If the response has N ring coefficients, then, for $t > 0$,

$$\Pr[\|\mathbf{z}\|_\infty > t] \le 2N \exp\left(-\frac{t^2}{2(V + Mt/3)}\right).$$

The same bound holds after centered coefficient reduction modulo q. The case $V = 0$ has deterministic zero response.

Proof. Fix u and put $X_a = (c_a * \mathbf{s}_a)_u$. Negation symmetry centers X_a. In the base family, support and sign averaging give $\mathbb{E}[X_a^2] \le \kappa_{2,a}^2 E_{a,u}^{\mathsf{loc}}/d_a$. Conditioning this nonnegative quantity on acceptance costs at most $1/\underline{\alpha}_a$, while the deterministic magnitude bound gives $\mathbb{E}[X_a^2] \le m_{a,u}^2$. For $0 < \theta < 3/M$, expansion of the exponential, centering, and $|\mathbb{E}[X_a^k]| \le \mathbb{E}[X_a^2] M^{k-2}$, together with $k! \ge 2 \cdot 3^{k-2}$ for $k \ge 2$, give

$$\mathbb{E}[e^{\theta X_a}] \le \exp\left(\frac{\theta^2 \mathbb{E}[X_a^2]}{2(1 - \theta M/3)}\right).$$

Multiply over the independent challenges and apply Chernoff with $\theta = t/(V + Mt/3)$. Apply the same argument to $-X_a$ and union-bound over the N output coefficients. Centered reduction cannot increase an absolute coefficient, proving the last claim.

For an embedded challenge subring, $E_{a,u}^{\mathsf{loc}}$ refers to the exact d_a-coordinate source fibre contributing to u, not the whole ambient ring. If $\|c_a\|_1 \le K_{1,a}$ for every accepted challenge, a uniform coefficient envelope B_a gives the simpler choices $m_{a,u} = K_{1,a} B_a$ and $E_{a,u}^{\mathsf{loc}} = d_a B_a^2$. All source envelopes must hold after fixing the preceding transcript.

K.2 From Response Bounds to Completeness

We give two ways to certify a uniform bound on $q_h(\tau)$. The first uses deterministic envelopes that cover every accepted challenge and every reachable source.

Corollary K.9 (Completeness from deterministic response envelopes). *Fix an admissible schedule. For every response stage h, response $r \in \mathcal{R}_h$, and reachable honest prefix τ, form the unreduced integer response as*

$$\mathbf{z}_{h,r} = \sum_{a \in \mathcal{Q}_{h,r}} c_{h,a} * \mathbf{s}_{h,r,a}(\tau).$$

Suppose the schedule gives bounds, uniform over these prefixes,

$$\|\mathbf{s}_{h,r,a}(\tau)\|_\infty \le B_{h,r,a}, \qquad \|\mathbf{s}_{h,r,a}(\tau)\|^2_{2,\mathrm{coef}} \le E_{h,r,a}.$$

For every accepted challenge, suppose also that $\|c_{h,a}\|_1 \le K_{1,h,a}$ and $\|c_{h,a}\|_{\mathsf{mul},2} \le \Gamma_{h,a}$. Define

$$U_{h,r} := \sum_a K_{1,h,a} B_{h,r,a}, \qquad D_{h,r} := \left(\sum_a \Gamma_{h,a} \sqrt{E_{h,r,a}} \right)^2.$$

If $T_{h,r} \ge U_{h,r}$ for every response and $S_{\max,h,r} \ge D_{h,r}$ whenever $p_{h,r} = 2$, then $q_h^{\mathsf{rsp}}(\tau) = 0$ for every reachable honest prefix. Consequently, any certified lower bound $\underline{a}_h \le a_h$ gives

$$\xi_h = (1 - \underline{a}_h)^{N_{\mathrm{try},h}}$$

in Theorem 10.9, without a modeled-response assumption.

Proof. Young's coefficient inequality and the triangle inequality give $\|\mathbf{z}_{h,r}\|_\infty \le U_{h,r}$. The multiplication-operator bound and the Euclidean triangle inequality give $\|\mathbf{z}_{h,r}\|^2_{2,\mathrm{coef}} \le D_{h,r}$, and centering modulo q preserves both bounds. Thus every successfully sampled challenge tuple satisfies every honest admission predicate, including those of the terminal. No independence among responses is required. It follows that $q_h(\tau) = 1 - a_h \le 1 - \underline{a}_h$; apply Corollary 10.10.

The envelopes may be obtained from the scheduled digit alphabets and source maps, without assuming that recursive digits are independent or uniformly distributed. Earlier challenge choices and successful nonce searches may therefore determine the sources adaptively, provided the same envelopes hold for every resulting honest prefix. Without an operator filter one may always use $\Gamma_{h,a} = K_{1,h,a}$. These conditions do not certify the optimized schedules derived from the response model: enlarging their bounds can change the digit depths, recursive witness dimensions, and Module-SIS widths.

Deterministic envelopes can be conservative. The second route instead assigns a failure budget to each encoding and squared-norm event and combines the resulting marginal bounds without assuming that the events are independent.

Corollary K.10 (Completeness from marginal response bounds). *Fix a response stage h, and let $p_{\mathsf{acc},h} \in (0,1)$ be its target lower bound on the conditional response-admission probability after successful challenge construction. For each response r, choose an encoding-failure allocation $\varepsilon^{\mathsf{enc}}_{h,r} > 0$. For each Euclidean response, also choose a squared-norm failure allocation $\varepsilon^2_{h,r} > 0$. Suppose*

$$\sum_{r \in \mathcal{R}_h} \left(\varepsilon^{\mathsf{enc}}_{h,r} + \mathbf{1}_{\{p_{h,r}=2\}} \varepsilon^2_{h,r} \right) \le 1 - p_{\mathsf{acc},h}. \tag{311}$$

For every reachable prefix τ, suppose the source-specific analysis gives

$$\Pr_{\boldsymbol{c}_h}[\neg \mathsf{Enc}_{h,r}(\mathbf{z}_{h,r}) \mid \tau, \boldsymbol{c}_h \ne \perp] \le \varepsilon^{\mathsf{enc}}_{h,r}. \tag{312}$$

For a dense response satisfying the conditional Rademacher-sign premises of Lemma K.3, with $N_{h,r}$ coefficients and uniform squared-weight bound $V_{h,r}$, it suffices to choose a symmetric threshold $Z_{h,r} \le T_{h,r}$ such that

$$Z_{h,r} \ge \sqrt{2V_{h,r}\log\left(\frac{2N_{h,r}}{\varepsilon_{h,r}^{\mathsf{enc}}}\right)}. \tag{313}$$

The one-hot tail of (306) gives the analogous root instantiation. For each Euclidean response, suppose additionally that for every reachable prefix

$$\mathbb{E}[\|\mathbf{z}_{h,r}\|_2^2 \mid \tau, \boldsymbol{c}_h \neq \perp] \le \mu_{h,r}$$

and set

$$S_{\max,h,r} \ge \frac{\mu_{h,r}}{\varepsilon_{h,r}^2}. \tag{314}$$

Then

$$q_h^{\mathsf{rsp}}(\tau) \le 1 - p_{\mathsf{acc},h}, \qquad q_h(\tau) \le 1 - a_h p_{\mathsf{acc},h} \tag{315}$$

for every reachable prefix. The expected number of nonce trials at that level, conditioned on reaching it, is at most $1/(a_h p_{\mathsf{acc},h})$, and nonce exhaustion contributes at most

$$(1 - a_h p_{\mathsf{acc},h})^{N_{\mathsf{try},h}} \tag{316}$$

to (156). No independence between different responses at the same level is required.

Proof. Equation (312) controls the canonical digit decomposition required on both norm routes. For a dense source it follows from Lemma K.3 and (313); for a canonical one-hot root it follows from Theorem K.5. Markov's inequality and (314) control the additional Euclidean event. The union bound and (311) give the pointwise conditional response bound. Equation (150) gives (315). Apply Corollary 10.10.

A response model may instead bound the joint event of Definition 10.5 directly (Appendix K.3).

K.3 A Response Model for Parameter Selection

Instead of certifying bounds, the planner may use a model to propose them and to estimate honest retry cost (Section 12.3). This subsection records the model's moment calculations and acceptance parameters. They do not establish the history-conditional completeness premise for the actual integer responses, which is stated separately in (340).

Acceptance Targets and Moment Estimates At level h, let $p_{\mathsf{acc},h} \in (0,1)$ be the target probability that all responses satisfy their bounds, conditioned on successful construction of the candidate challenge tuple. For each response r, let $p_{h,r}^{\mathsf{rsp}}$ be its target probability of satisfying its complete honest admission predicate. Unless a joint analysis of several responses is provided, the policy requires

$$\sum_{r \in \mathcal{R}_h} \left(1 - p_{h,r}^{\mathsf{rsp}}\right) \le 1 - p_{\mathsf{acc},h}. \tag{317}$$

For a single recursive response, $p_{h,r}^{\mathsf{rsp}} = p_{\mathsf{acc},h}$. Certified analyses may instead instantiate the marginal failure allocation of Corollary K.10. Let $\rho_{\mathsf{model},h} \ge 1$ be an envelope for model error, and let $\mathsf{Round}_{\mathsf{mom},h} : \mathbb{R}_{\ge 0} \to \mathbb{R}_{\ge 0}$ be a monotone upward-rounding map. These are planning parameters; the schedule records the bound derived from them and the allowed nonce count $N_{\mathsf{try},h}$.

For challenge coordinate u, let $\underline{\alpha}_{h,u}$ be the certified lower bound on the accepted fraction of its base family and let $L_{h,u}$ be its draw limit. The planner uses the certified challenge-construction success probability

$$\underline{a}_{h,u} := 1 - (1 - \underline{\alpha}_{h,u})^{L_{h,u}}, \qquad \underline{a}_h := \prod_{u \in U_h} \underline{a}_{h,u} \le a_h. \tag{318}$$

An unfiltered coordinate has $\underline{\alpha}_{h,u} = 1$. The sampler also satisfies the polynomial bounds and random-oracle query-accounting conditions of Definition 9.2, ensuring the compatibility required by Lemma 10.2.

For one recursive source, define the component set

$$\mathsf{RespPart} := \{\mathsf{z}, \mathsf{e}, \mathsf{t}, \mathsf{r}, \mathsf{cmp}\},$$

one for each kind of segment in (76): response digits $\hat{\mathbf{z}}$, partial-opening digits $\hat{\mathbf{e}}$, inner-image digits $\hat{\mathbf{t}}$, quotient digits, and compression digits. With d_{pack} the dimension of the packing ring, we record the moment estimates separately for each component as

$$\mathsf{M} = \left(d_{\mathsf{pack}}; (\mu_\chi, v_\chi^{\mathsf{full}}, v_\chi^{\mathsf{local}})_{\chi \in \mathsf{RespPart}}\right). \tag{319}$$

Here μ_χ models the total squared coefficient norm of component χ. The statistics v_χ^{full} and v_χ^{local} are inputs to the model's peak-column estimate: the former uses an average over a complete packing ring, whereas the latter estimates a largest coefficient second moment when a strict local subring is exposed. They are planning surrogates, not certified maxima of the ring-coefficient second moments. The modeled total squared norm is $\mu(\mathsf{M}) := \sum_\chi \mu_\chi$. Every moment estimate is rounded upward through $\mathsf{Round}_{\mathsf{mom},h}$.

Rigorous root states. Let a balanced source contain L base-field coefficients, and let digit plane j have absolute bound $B_{\mathsf{dig},j}$. Its initial state is

$$\mu_{\mathsf{z}} = L \sum_{j=0}^{\delta-1} B_{\mathsf{dig},j}^2, \qquad v_{\mathsf{z}}^{\mathsf{full}} = v_{\mathsf{z}}^{\mathsf{local}} = \max_j B_{\mathsf{dig},j}^2. \tag{320}$$

For a unit one-hot source, the canonical layout determines the number H of unit coefficients after packing exactly, giving

$$\mu_{\mathsf{z}} = H, \qquad v_{\mathsf{z}}^{\mathsf{full}} = v_{\mathsf{z}}^{\mathsf{local}} = 1. \tag{321}$$

Thus neither root state assumes a distribution on the incoming witness. The corresponding coefficient tails are proved in Lemma K.3 and Theorem K.5.

For a uniformly distributed centered base-b digit, define

$$m_2^{\mathsf{unif}}(b) := \frac{1}{b} \sum_{a=-b/2}^{b/2-1} a^2 = \frac{b^2+2}{12}. \tag{322}$$

When a public setup value is modeled as uniform in the field, its digit component sums (322) over the complete planes and uses the actual residual width of the final plane.

Peak-column functional. Let N_{src} be the number of source coefficients that can enter one response column. For $\star \in \{\mathsf{full}, \mathsf{local}\}$, define

$$\mathsf{Peak}^\star_{N_{\mathsf{src}}}(\mathsf{M}) := \max\left\{ \sum_{\chi \in \mathsf{RespPart}} n_\chi v_\chi^\star : 0 \le n_\chi \le \frac{\mu_\chi}{v_\chi^\star}, \sum_\chi n_\chi \le N_{\mathsf{src}} \right\}, \tag{323}$$

with a zero component omitted. Equivalently, fill the N_{src} positions from the largest component moments downward, using at most the component's total squared norm. This keeps disjoint witness components from each being counted at the full column capacity.

Propagating Moments through a Fold Consider a response with N ring coefficients, challenge squared norm κ_2^2, ring dimension d, and B_{col} source blocks per response column. When only a slice or response chunk is folded, M below denotes the state restricted to the exact source spans assigned to that response. The model first sets

$$\widehat{\mu}_{\mathsf{resp}} := \kappa_2^2 \mu(\mathsf{M}), \tag{324}$$

$$\widehat{v}_{\mathsf{resp}} := \frac{\kappa_2^2}{d} \mathsf{Peak}^\star_{dB_{\mathsf{col}}}(\mathsf{M}), \tag{325}$$

where $\star = \mathsf{full}$ if the exposed ring equals d_{pack} and $\star = \mathsf{local}$ otherwise.

The response is then digit-decomposed. Let $\lfloor x \rceil$ denote nearest integer rounding and let $\mathrm{cent}_b(x)$ be the centered residue of an integer modulo b, and define the rounded-normal digit moment

$$\mathsf{GDig}(\sigma, b) := \mathbb{E}_{X \leftarrow \mathcal{N}(0,\sigma^2)} \left[\mathrm{cent}_b(\lfloor X \rceil)^2 \right]. \tag{326}$$

For response base b and digit depth δ_f, the recursive z component is modeled by

$$\mu'_{\mathsf{z}} := N \sum_{j=0}^{\delta_f - 1} \mathsf{GDig}\left(\sqrt{\widehat{\mu}_{\mathsf{resp}}/N}/b^j, b \right), \tag{327}$$

$$v'_{\mathsf{z}} := \max_{0 \le j < \delta_f} \mathsf{GDig}\left(\sqrt{\widehat{v}_{\mathsf{resp}}}/b^j, b \right). \tag{328}$$

These two equations are the Gaussian-inspired step of the model; the rigorous root bounds above do not use them.

The e, t, and r components use the uniform digit moment of (322) at their exact component-specific scalar counts and digit depths. A negative-binary compression coordinate has modeled second moment $1/2$, so a compression span of length L_{cmp} contributes $L_{\mathsf{cmp}}/2$. These component calculations use the actual witness layout rather than one scale factor applied to its total length.

Finally, the two-bank packing map (22) has integer-lift squared norm

$$\|\mathbf{c}\|_2^2 = \sum_t (a_{t,0}^2 + b_{t,0}^2) + 2 \sum_t \sum_{j=1}^{k-1} (a_{t,j}^2 + b_{t,j}^2). \tag{329}$$

Thus exchangeable second moments across extension coordinates give the expected total squared-norm multiplier $(2k-1)/k$; no independence is needed because the cross terms cancel between the two packed coefficient positions. The planner applies this factor to the total squared norm and to the full-ring averaged statistic, and approximates the local statistic:

$$\mu_\chi^{\mathsf{pack}} = \frac{2k-1}{k} \mu_\chi, \qquad v_\chi^{\mathsf{full,pack}} = \frac{2k-1}{k} v_\chi^{\mathsf{full}}, \qquad v_\chi^{\mathsf{local,pack}} := \begin{cases} v_\chi^{\mathsf{local}}, & k = 1, \\ 2 v_\chi^{\mathsf{local}}, & k > 1. \end{cases} \tag{330}$$

The local entry is a model approximation, not a bound on every packed coordinate. For an overlapping coefficient pair A, B, the exact second moments are

$$\mathbb{E}[(A \pm B)^2] = \mathbb{E}[A^2] + \mathbb{E}[B^2] \pm 2\mathbb{E}[AB]. \tag{331}$$

The model assumes $\mathbb{E}[AB] = 0$: if both input second moments are at most v, this gives the local bound $2v$. Exchangeability alone does not imply $\mathbb{E}[AB] = 0$, and independence does not imply it for nonzero-mean inputs: independent uniform digits in $\{-1, 0\}$ have second moment $1/2$ but $\mathbb{E}[(A+B)^2] = 3/2$. Without a cross-moment assumption, Cauchy–Schwarz gives the conservative bound $4v$, so a certified coordinatewise propagation must retain the cross moments or use that bound. Rounding the five packed components upward gives the successor model state, whose use to select bounds remains subject to the history-conditional admission premise (152).

From Moments to Scheduled Bounds For response r with N ring coefficients, let $\widehat{\mu}_{h,r}$ be the squared norm predicted by the state transition, and define its inflated squared norm and marginal-variance parameters by

$$\overline{\mu}_{h,r} := \rho_{\mathsf{model},h} \widehat{\mu}_{h,r}, \tag{332}$$

$$\widehat{v}_{h,r} := \rho_{\mathsf{model},h} \max\{\widehat{\mu}_{h,r}/N, \widehat{v}_{\mathsf{resp},h,r}\}. \tag{333}$$

For every reachable transcript prefix, the model replaces the response by a centered jointly Gaussian surrogate $\mathbf{X}_{h,r}$ satisfying

$$\mathbb{E}[\|\mathbf{X}_{h,r}\|_2^2 \mid \mathsf{tr}_{<h}, \boldsymbol{c}_h \neq \perp] \le \overline{\mu}_{h,r}, \qquad \max_u \mathrm{Var}[X_{h,r,u} \mid \mathsf{tr}_{<h}, \boldsymbol{c}_h \neq \perp] \le \widehat{v}_{h,r}. \tag{334}$$

Let Φ be the standard normal cumulative distribution function. For a candidate digit depth δ, let $T_{h,r}(\delta)$ be the positive endpoint of its canonical representable interval and define

$$p_{h,r}^{\mathsf{box}}(\delta) := \left(2\Phi\left(\frac{T_{h,r}(\delta)}{\sqrt{\widehat{v}_{h,r}}}\right) - 1\right)^N. \tag{335}$$

If $\widehat{v}_{h,r} = 0$, define $p_{h,r}^{\mathsf{box}} = 1$ for every nonnegative threshold: the Gaussian surrogate is identically zero. The terminal creates no response-digit witness. If the implementation imposes an encoding limit there, choose $T_{h,r}(\delta)$ so that every vector in $[-T_{h,r}(\delta), T_{h,r}(\delta)]^N$ fits the selected encoding; the box event is then sufficient for encodability. If the squared-norm bound alone guarantees encodability, take $T_{h,r} = +\infty$ and $p_{h,r}^{\mathsf{box}} = 1$ in the following bounds. Repeated application of the Gaussian correlation inequality [89] to the coordinate slabs of $\mathbf{X}_{h,r}$ gives

$$\Pr[\|\mathbf{X}_{h,r}\|_\infty \le T_{h,r}(\delta) \mid \mathsf{tr}_{<h}, \boldsymbol{c}_h \ne \perp] \ge p_{h,r}^{\mathsf{box}}(\delta). \tag{336}$$

For a squared-norm bound $S > \overline{\mu}_{h,r}$, Markov's inequality gives the ball score

$$p_{h,r}^{\mathsf{ball}}(S) := 1 - \frac{\overline{\mu}_{h,r}}{S}. \tag{337}$$

If $\overline{\mu}_{h,r} = 0$, the surrogate is identically zero; define $p_{h,r}^{\mathsf{ball}}(0) = 1$ as well. The cube and Euclidean ball are both centrally symmetric and convex. Applying the Gaussian correlation inequality once more gives the joint surrogate bound

$$\begin{aligned}\Pr\big[\|\mathbf{X}_{h,r}\|_\infty \le T_{h,r}(\delta) \ \wedge\ \|\mathbf{X}_{h,r}\|_2^2 \le S \mid \mathsf{tr}_{<h}, \boldsymbol{c}_h \ne \perp\big] \\ \ge p_{h,r}^{\mathsf{G}}(\delta, S) := p_{h,r}^{\mathsf{box}}(\delta) p_{h,r}^{\mathsf{ball}}(S).\end{aligned} \tag{338}$$

No independence between the coordinates, or between the box and ball events, is used in this calculation.

For a target $p_{h,r}^{\mathsf{rsp}}$ and a depth satisfying $p_{h,r}^{\mathsf{box}}(\delta) > p_{h,r}^{\mathsf{rsp}}$, the least bound implied by the surrogate score is

$$S_{h,r}^{\mathsf{G}}(\delta) := \left\lceil \frac{\overline{\mu}_{h,r}}{1 - p_{h,r}^{\mathsf{rsp}}/p_{h,r}^{\mathsf{box}}(\delta)} \right\rceil. \tag{339}$$

The planner enumerates the admissible digit depths and closes the complete downstream cost of each pair $(\delta, S_{h,r}^{\mathsf{G}}(\delta))$. On the coefficient route it instead requires only $p_{h,r}^{\mathsf{box}}(\delta) \ge p_{h,r}^{\mathsf{rsp}}$.

The actual recursive response is integer valued and is not proved to be Gaussian. For a model-derived response, the schedule therefore invokes the joint history-conditional premise

$$\Pr[\mathsf{HAdm}_{h,r}(\mathbf{z}_{h,r}) \mid \mathsf{tr}_{<h}, \boldsymbol{c}_h \ne \perp] \ge p_{h,r}^{\mathsf{rsp}} \tag{340}$$

for every reachable prior transcript. Equation (338) motivates this premise but does not prove it. For an integer response, the deterministic condition $S_{\max,h,r} < (T_{h,r} + 1)^2$ is an alternative sufficient route: in that case the squared-norm event itself implies encodability.

Proposition K.11 (From per-response model premises to stage completeness). *Suppose* (340) *holds for every response at level h, with response targets satisfying* (317)*. Then the level satisfies* (152) *with $\widehat{\delta}_h^{\mathsf{rsp}} = 1 - p_{\mathsf{acc},h}$ and* (151) *with $\delta_h = 1 - \underline{a}_h p_{\mathsf{acc},h}$. Consequently, one complete nonce attempt succeeds with conditional probability at least $\underline{a}_h p_{\mathsf{acc},h}$, and allowing $N_{\mathsf{try},h}$ fresh nonce attempts makes the exhaustion probability at most $(1 - \underline{a}_h p_{\mathsf{acc},h})^{N_{\mathsf{try},h}}$.*

Proof. Apply a union bound to the complete response-admission events. Equation (317) gives the history-conditional one-shot claim; no independence between responses is required. Fresh random-oracle nonces, the bounded-sampler guarantee (318), and Lemma 10.7 give the exhaustion bound.

Comparing Coefficient and Squared-Norm Bounds The distinction between coefficient-wise and squared-norm bounds remains even within a single response model. Let N_A be the number of response coefficients, and let Z_∞ and $S_{\max}$ be their respective bounds. Suppose, for intuition, that the ring coefficients are independent Gaussians with scale σ. A coefficient bound that fails with probability at most ε_∞ has the approximate size

$$Z_\infty^2 \approx 2\sigma^2 \log\left(\frac{2N_A}{\varepsilon_\infty}\right), \tag{341}$$

whereas the total squared norm concentrates around $N_A\sigma^2$. A corresponding squared-norm bound has the approximate form

$$S_{\max} \approx \sigma^2 \left(N_A + 2\sqrt{N_A \log(1/\varepsilon_2)} + 2\log(1/\varepsilon_2)\right). \tag{342}$$

When N_A dominates $\log(1/\varepsilon_2)$, these estimates give

$$\frac{N_A Z_\infty^2}{S_{\max}} \approx 2\log\left(\frac{2N_A}{\varepsilon_\infty}\right). \tag{343}$$

The box loses a logarithmic factor in squared radius, or its square root in radius; the ℓ_2 bound rules out many coordinates approaching the coefficient bound at once.

K.4 A Certified Alternative for Digit Squared Norms

The Gaussian digit transition can be replaced by a pointwise certificate. Let $I_{b,\delta}$ be the exact integer interval representable by the canonical balanced base-b, depth-δ decomposition, and write

$$g_{b,\delta}(x) := \sum_{j=0}^{\delta-1} \mathrm{digit}_{b,j}(x)^2.$$

For a rational $\theta \geq 0$, define the finite maximum

$$A_{b,\delta}(\theta) := \max_{x \in I_{b,\delta}} \left(g_{b,\delta}(x) - \theta x^2\right).$$

Every representable vector $\mathbf{z}$ of length N and squared norm at most S then satisfies

$$\sum_{i=0}^{N-1} g_{b,\delta}(z_i) \leq N A_{b,\delta}(\theta) + \theta S. \tag{344}$$

Indeed, the defining inequality for $A_{b,\delta}(\theta)$ holds at each integer coefficient, and summing it proves the claim. The certificate therefore remains valid in the presence of correlations, carries, and earlier nonce selection. It can be checked by exact enumeration of the interval or by a certified digit recurrence, and a finite search over rational θ gives certified upper bounds without a model of the digit distribution.

The resulting bound need not fit the next fold's existing budget. Carry patterns allowed by the preceding coefficient and norm bounds can already contribute too much digit squared norm before the opening, matrix-image, and compression segments are added. Preserving optimized parameters therefore requires information about the actual honest source distribution beyond these bounds, or a proved bound on the probability of exceptional source histories. Restricted source windows and coefficient packing also require their exact multiplicities after packing. Using this certificate for one entry does not remove the model premise from a schedule whose response bounds are model-derived.

Conditions for certifying the optimized parameters. The quantitative targets are history-conditional squared norms and local second moments for the individual source segments, together with a covariance bound for the actual filtered challenges; the accepted-fraction bound of Lemma K.7 can be much larger than the modeled conditional mean. A proof may instead establish the source properties outside a small exceptional set of honest histories, charging its probability separately from nonce exhaustion. Parameter search can still use a model to propose bounds, while an independent check verifies each level's source premise, squared-norm propagation, conditional tail, and joint retry budget. Such a check would remove the modeled-response premise only for schedules whose proposed parameters satisfy all these conditions.

K.5 Choosing the Response Digit Depth

We now explain how the schedule chooses the response digit depth δ_f of Section 5.5. Suppress the group index, and let b be the response base and $b^\star \geq b$ the certified range base. The honest prover uses the canonical base-b decomposition, whose exact representable interval is

$$[-M_f, T_f], \qquad M_f := \frac{b}{2} \cdot \frac{b^{\delta_f} - 1}{b-1}, \qquad T_f := \left(\frac{b}{2} - 1\right) \frac{b^{\delta_f} - 1}{b-1}. \tag{345}$$

An honest response is encodable by δ_f canonical digits exactly when its coefficients lie in this interval, so a symmetric honest-response threshold used to choose δ_f must be at most T_f. The certified envelope (77) is a different bound: it holds for every accepted response, and when $b < b^\star$ it is strictly larger than the canonical representable interval, so it cannot justify honest encodability.

The threshold comes from Lemma K.3 and Theorem K.5 for dense and canonical one-hot root sources, when the accepted challenge distribution satisfies their hypotheses, and from the response model of Appendix K.3 for recursive responses. In every case it is rounded up to the least digit depth whose canonical interval (345) contains it, and the wider envelope (77) is then derived from that depth for security sizing. The retry allowance $N_{\mathsf{try},h}$ is an independent schedule parameter, accounted for by Theorem 10.9.

L Implementation Details

This appendix collects layout, encoding, and sampling details that an implementation must fix but that the security analysis does not use.

L.1 Setup and Commitment Layout

This subsection fixes layout and encoding details of the setup and commitment of Sections 4.3 and 4.5.

Setup draw width. Section 4.3 requires each setup coefficient in (36) to be within statistical distance $q\,2^{-w}$ of uniform, with aggregate bias $N_{\mathsf{setup}}\,q\,2^{-w}$. For an allocated setup error $\delta_{\mathsf{setup}} > 0$, a sufficient draw width for this aggregate bound is

$$w \geq \lceil \log_2 N_{\mathsf{setup}} + \log_2 q + \log_2(1/\delta_{\mathsf{setup}}) \rceil .$$

A width of twice the modulus bit length does not by itself meet this target, so the setup profile records the actual sampling map and width and checks the bound for the largest admitted prefix. Exact rejection sampling has zero coefficient bias.

Outer-commitment address permutation. The R_{q,d_B} view $\hat{\mathbf{t}}^{(B)}_{p,i}$ in (44) is a fixed permutation of the opening digits $\hat{\mathbf{t}}_i$. Writing $s = d_A/d_B$ and $\delta = \delta_{\mathsf{open}}$, the coefficient address changes as

$$(a\delta + u)d_A + y d_B + k \quad \longmapsto \quad ((as + y)\delta + u)d_B + k.$$

Here a indexes the $\mathbf{A}$ row, $y \in [s]$ the native subcolumn, $u \in [\delta]$ the digit, and $k \in [d_B]$ its coefficient.

Binary digit placement. Each compression stage of Section 4.5 decomposes an image with s coefficients into $\delta = \lceil \log_2 q \rceil$ digits per coefficient. Digit u of coefficient v is placed at position $v + su$, so that the least significant digits of all coefficients come first, then their next digits, and so on. This sequence is packed into consecutive input ring elements of $\mathbf{F}_j$, with zeros filling the last element. The gadget map $\mathbf{G}_{\mathcal{A}_{F,j},\cdot}$ reverses this packing: it sends a digit sequence $\boldsymbol{\xi}$ to the coefficient vector whose vth entry is $\sum_{u=0}^{\delta-1} 2^u \xi_{v+su}$ modulo q. The $\mathbf{H}$-chain uses the same placement.

Payload encoding. The payload (46) is encoded as follows. At depth zero, it is a vector of $S_B n_B d_B$ field elements in slice, matrix-row, and coefficient order. At positive depth, it is a vector of $n_{F,L_F} d_{F,L_F}$ field elements in ring-row-major coefficient order. The field uses its fixed-width canonical encoding in both cases. The payload carries no shape or length header; the transcript-bound schedule determines both. The evaluated rank-one compression schedules specialize the positive-depth order to $c_0, \ldots, c_{d_{F,L_F}-1}$.

Cost of slicing. Slicing the outer commitment (45) bounds the **B** setup view at $n_B m_B d_B$ field coefficients and sizes its Module-SIS instance at the matrix width m_B. In exchange, the stacked image has $S_B\, n_B$ ring elements rather than n_B. This larger image enlarges the input to the **F**-chain with compression, or the payload itself without compression. It also enlarges the outer-consistency rows, since each slice contributes n_B of them to the common relation, and the public weight evaluation. These effects enter the transition cost and successor shape of Section 12.2. The public weights of the slices can be combined before reading each setup coefficient, so slicing increases the weight-evaluation work without replicating the setup matrix; Appendix C.4 gives this contraction and its cost.

L.2 Fold Witness Layout and Encodings

This subsection fixes the witness addresses and zero alignment of Section 5.5 and the terminal response encoding of Section 8.2.

Digit addresses. The witness is a vector of base-field coefficients, and each segment keeps its own ring dimension (Section 5.5). For one group, suppress its index. To store a value with base-field width D in dimension $d \mid D$, first split it into $s = D/d$ consecutive blocks, then decompose each block. For δ digits per block, the local address is

$$\mathrm{pos}(j, y, u, k) := \big((js + y)\delta + u\big)d + k, \qquad y \in [s], \quad u \in [\delta], \quad k \in [d]. \tag{346}$$

Thus the coefficient coordinate changes first, then the digit, then the native block. For partial opening evaluations, j enumerates claim/block pairs and $(D, d) = (d_{\mathsf{op},g}, d_D)$ from (68). For inner images, j enumerates claim/block/**A**-row triples and $(D, d) = (d_A, d_B)$. A response has no claim axis. Its coefficient k at position x, input digit u_{com}, and response digit u_{fold} has address

$$\mathrm{pos}_{\hat{\mathbf{z}}}(x, u_{\mathsf{com}}, u_{\mathsf{fold}}, k) := \big(\delta_f(\delta_{\mathsf{com}} x + u_{\mathsf{com}}) + u_{\mathsf{fold}}\big)d_A + k. \tag{347}$$

Offsets are sums of preceding segment lengths. Compression digits retain the digit-plane order and ring-completion padding of Section 4.5. Digit counts are the actual counts chosen by the schedule, without rounding to powers of two.

Quotients retain their row's native dimension and coefficient field. If row r has dimension d_r and p_r base-field coordinates per coefficient, its digit u, field coordinate t, and ring coefficient k occupy the local position

$$\mathrm{pos}_{\hat{\mathbf{Q}}_r}(u, t, k) := (up_r + t)d_r + k. \tag{348}$$

The schedule fixes the quotient-digit decomposition.

Zero alignment. Equation (76) suppresses scheduled zero alignment: the compression suffix starts on the common relation-coefficient block boundary, and each compression stage starts on a boundary of its largest ring dimension. The compression suffix ends on a boundary of the largest group d_A when there are several groups, or of the common relation block when there is one. The common block size is the smallest polynomial-modulus dimension among the noncompression rows. Together with the compression maps' ring-completion padding, these zeros are part of the committed vector. A final zero suffix completes the successor rings and extends the vector to the Boolean domain. Within $\mathbf{w}^{(h)}_{\mathsf{main}}$ and $\hat{\mathbf{Q}}_{\mathsf{main}}$, segments are concatenated without padding to the largest ring dimension. The order follows the relations and compression stages, not a global sort by dimension.

Terminal response encoding. The terminal reveals $\mathbf{z}$ and checks its scheduled bound directly (Section 8.2). On the Euclidean route, the norm check imposes no additional coefficient bound. A signed Rice encoding can use a byte budget derived from the squared-norm bound: for N_z coefficients and bound S, the inequality $\sum_i |z_i| \le \sqrt{N_z S}$ bounds the unary portion of the Rice code. Such a budget introduces no separate encoding-admission event.[18] The encoding preserves the segment order used by the committed folds; bounded responses use the scheduled prefix code and dense segments use fixed-width coefficients as specified in the Akita Book [65].

[18] An implementation may instead impose a tighter encodability limit; response admission then also checks that the encoding fits this limit.

L.3 Challenge Sampling

The three-step challenge rule of Section 3.2 is uniform on $\mathcal{C}$. It can be implemented with partial Fisher–Yates shuffles, using rejection sampling to draw each exact uniform integer from the required interval, followed by one independent sign bit for each of the s support positions. In the noninteractive protocol these draws come from a domain-separated XOF stream. Because of rejection, the number of bits read for an interval of size n is variable rather than exactly $\lceil \log_2 n \rceil$; the Akita Book specifies the byte-level procedure [65].

M PCS Benchmark Scope and Parameters

The standalone comparison measures native PCS openings at equal nominal field capacity. This appendix specifies which evaluation problems the rows represent and which reductions are excluded from the measurements; Appendix N describes the measurement procedure.

Variable counts and equality weights. For a multilinear polynomial over $\mathbb{F}$ with $N = 2^\ell$ evaluations on the Boolean cube, the nominal payload is $B = N \log_2 |\mathbb{F}|$. The input values are sampled over the native field, rather than restricted to 0 and 1. This capacity is measured before Akita decomposes field elements into small digits for its lattice commitment; the digits do not redefine the polynomial or its number of variables. An opening point has ℓ coordinates in the opening field $\mathbb{E}$, and evaluation weights the 2^ℓ input values by $\widetilde{\mathsf{eq}}(r, x)$. Changing the coefficient field at fixed B therefore changes both the point dimension and the number of values processed.

For a nominal target $B = 2^b$, the approximately 32-, 64-, and 128-bit fields use $N = 2^{b-5}, 2^{b-6}, 2^{b-7}$, respectively. At $B = 2^{31}$, their multilinear polynomials have 26, 25, and 24 variables. The two-coordinate difference between the 32- and 128-bit profiles corresponds to a factor of four in the number of equality weights. Akita's opening fields have extension degrees 4, 2, and 1 over these three base fields, so each opening-field element has approximately 128 bits. A dense array with one such element per Boolean evaluation would thus be four times larger for the 32-bit profile, even though the input coefficient arrays have the same nominal capacity. These counts describe the workload; they do not assert that every prover allocates such an array.

Opening-field sizes also differ across implementations. Plonky2 uses a quartic extension of Goldilocks; Plonky3 FRI and STIR use a degree-5 extension of KoalaBear and Plonky3 WHIR a degree-8 extension; SP1 uses a degree-4 extension of KoalaBear; and WorldFnd WHIR uses a cubic extension of Goldilocks. Their opening-field sizes are approximately 256, 155, 248, 124, and 192 bits, respectively. Binius64 and Flock open over $\mathbb{F}_{2^{128}}$.

FRI/STIR measure univariate batches at large inputs. Plonky3 FRI and STIR use a different evaluation interface from the multilinear PCSs. Their KoalaBear field supports power-of-two transform domains only up to 2^{24}. At the measured code rate 1/2, a single native univariate instance can therefore have at most 2^{23} input evaluations. The adapter distributes larger inputs among columns of a matrix with height 2^{23} and opens every column as a separate univariate polynomial at one shared extension-field point. The resulting workloads are:

Input bits	2^{27}	2^{29}	2^{31}	2^{33}	2^{35}
Field elements	2^{22}	2^{24}	2^{26}	2^{28}	2^{30}
Values per polynomial	2^{22}	2^{23}	2^{23}	2^{23}	2^{23}
Polynomials	1	2	8	32	128

At 2^{35} bits, the adapter thus proves 128 univariate evaluation claims, not one opening of a 30-variable multilinear polynomial. No reduction from that multilinear claim to the batched interface is included. Batching shares commitment and proof work, but queried input rows contain column values and the evaluation vector has one entry per polynomial. Opening-proof sizes exclude this separately reported evaluation vector; total communication includes it. The effect on runtime and proof size cannot be attributed to input capacity alone.

The transform limit belongs to the selected configuration, not to FRI or STIR in general. We report these native batched-univariate workloads rather than interpreting their timings as costs for the multilinear statement.

Packed openings and omitted tensor-algebra reductions. The Binius64 and Flock adapters directly measure openings of multilinear polynomials over $\mathbb{F}_{2^{128}}$. Binius64 generates native field elements; Flock generates the same packed-field representation of its nominal bit input. For a target of 2^b bits, both measured polynomials have 2^{b-7} field elements and $b-7$ variables. Neither adapter measures the tensor-algebra reduction needed to convert an arbitrary evaluation claim about an original bit or binary-subfield polynomial into the packed-field claims. This is the type of reduction discussed in Section 3.7, often called ring switching in the binary-field PCS literature; it is distinct from our quotient-lift ring switching.

For example, packing 2^b bits in groups of 128 changes the table length from 2^b to 2^{b-7}. An arbitrary opening of the original b-variable bit polynomial is not simply an opening of this $(b-7)$-variable polynomial: the original query weights vary within each packed group. The omitted reduction must account for those weights. Its prover and verifier time, additional communication, and working memory are excluded from the reported Binius64 and Flock results. The equality-weight work required by their native packed-field openings remains included in the adapters' opening measurements.

Timing boundaries and input preparation. For SP1, the harness separately computes the expected evaluation to check correctness, outside the timers; the opening timer includes producing the evaluation returned with the proof. This avoids charging the prover for a second evaluation used only by the benchmark to check its answer.

Plonky3 WHIR timings begin after construction of its native witness layout. For the measured single-polynomial input, this constructor duplicates the input without changing its order. We retain this timing boundary to measure commitment given the prepared native witness. The copy is an implementation choice, not an inherent WHIR operation; at 2^{35} nominal bits it creates an additional 4 GiB array. Its construction time is excluded, while its allocation contributes to process peak RSS. Its timing contribution has not been measured separately.

The WorldFnd WHIR adapter transfers ownership of its input vector to the prover buffer and borrows that buffer for evaluation. It therefore retains a single witness allocation rather than a redundant input copy. These measurements use that adapter. Claim evaluation remains inside the opening timer; the ownership transfer leaves the protocol parameters and proof-size accounting unchanged.

Security and parameters. Table 14 records the measured configurations. Targets follow each implementation's analysis and are not a common end-to-end security guarantee. Every hash row runs in unique decoding or in the Johnson regime, and meets its stated target under the bound proven for that regime; no row relies on proximity gaps up to capacity or on a conjectured FRI query bound. Binius64, Flock, WorldFnd, and SP1 use their product or default profiles unchanged. Where an upstream FRI, STIR, or WHIR profile reaches its target only under a conjecture, we keep its shape and take the regime and query count from a proven bound: the Plonky3 rows use the library's own soundness calculator, and the Plonky2 row uses the Johnson-regime FRI bound of [17, Theorem 8.3] over a quartic challenge field, since the library provides no calculator. Within the proven regime, rate, folding arity, and proof-of-work bits were chosen by measured prover time, with proof size as the tie-breaker. In the STIR row, MCA means *mutual correlated agreement*.

Table 14: Security accounting and parameter provenance for the hash-PCS comparison. RBR denotes round-by-round soundness, and UDR denotes unique decoding.

Scheme	Configuration in current measurements	Parameter source
Akita	128-bit Module-SIS and 128-bit classical-ROM transcript target; planner-selected schedules	Native implementation
Plonky2 FRI	Rate 1/2, fold 16, 169 queries, 16 work bits, quartic challenge field; 100-bit Johnson-regime bound	Derived from [17, Theorem 8.3]
Plonky3 FRI	Rate 1/2, fold up to 8, 169 queries, 16 query and 10 batching work bits; 100-bit Johnson-regime bound	Upstream benchmark profile with the query count from the library's calculator
Plonky3 STIR	Rate 1/2, fold 16 throughout, at most 16 work bits per phase; 100-bit Johnson/MCA target	Upstream benchmark profile with the Johnson bound
Plonky3 WHIR	128-bit RBR target, Johnson bound at every size; rate 1/2, degree-8 challenge field	Upstream benchmark profile with the Johnson bound
Binius64	Rate 1/2, SHA-256; 96-bit UDR query target	Product default
Flock	Default `Fast`: rate 1/2, Johnson and two OOD checks, SHA-256; 128-bit RBR target	Native implementation
WorldFnd	128-bit RBR target, Johnson bound; rate 1/2, fold 4, BLAKE3	CLI defaults

Scheme	Configuration in current measurements	Parameter source
SP1	100-bit target; rate 1/4, 124 queries, 16 work bits, stacking height 21	Product default

N Benchmark Methodology and Detailed Measurements

The measurements in Section 13 use the native interfaces and parameter profiles described in Appendix M. This appendix records their software provenance, measurement procedure, and communication accounting. The lattice results appear in Table 5 and Fig. 9; the hash results appear in Table 6 and Fig. 10, and Table 16 gives them for both prover thread counts and all input sizes, with communication, memory, and reusable preprocessing.

N.1 Software and Measurement Procedure

The `v0.1.0` release contains 16 workspace crates and approximately 248,000 lines of Rust across the libraries, tools, tests, benchmarks, and examples [67].[19]

The standalone Akita implementation is publicly available [66], with planner-selected dense schedules for $p_{32} = 2^{32} - 99$, $p_{64} = 2^{64} - 59$, and $p_{128} = 2^{128} - 2^{32} + 22537$. The direct and recursive setup-offload configurations share this revision. Planning and independent validation precede the measured online operations. Schedules using model-based response bounds retain the completeness premise of Section 12.3; passing the validator does not replace that premise with a proved honest-acceptance bound.

Both standalone comparisons use the observations supplied in the public benchmark artifact [69]. Each displayed value is the median of ten measured observations after one warmup for its scheme, input size, and thread count. Speedups are computed from unrounded medians. Each observation runs in a fresh process. Greyhound uses revision `672e7410`, with the security-parameterized sparse-ternary reference and contextual wire encoding described in Section 13. RoKoKo uses revision `5caba472`, which includes the radix-recomposition repair `ba83b317`. The current security qualifications are described in Appendix J.2; these measurements do not characterize the earlier implementation error. Hachi has no implemented end-to-end PCS in its public artifact and Grand Danois has no public runtime implementation, so neither is a timing baseline.

Standalone builds enable native CPU instructions. Akita uses Rust 1.95; RoKoKo uses nightly Rust with its native parameter features; Greyhound uses `-O3 -flto=auto -march=native`. The lattice harness disables RoKoKo's optional parallel-commitment feature and Greyhound's selected parallel kernels. Commitment, opening, and verification therefore use one thread in this comparison. The hash comparison measures one- and eight-thread proving, with one-thread verification. Hardware and process-memory limits are given in Section 13.

Measurement cohorts. All measurements were recorded on one machine in separate runs per scheme group. The lattice file contains 220 observations, including 20 warmups. Unsupported and out-of-memory rows are explicit observations rather than successful measurements; a cell marked n/m was not measured under the profile of Table 14. The artifact's manifest records the source identities and result-file digests of each measurement cohort. The lattice comparison has separate Akita, Greyhound, and RoKoKo runs; the hash comparison has one run for the six Akita field/offload profiles and separate runs for the baselines. Each record carries its source identities, build command, and executable and lockfile digests. The recorded harness identities carry a modified-tree marker with a digest of the modification; vendor checkouts are identified through dependency pins and executable digests. Table 15 lists the implementation revisions of the hash baselines; Appendix M specifies their parameter profiles and timing boundaries.

N.2 Inputs and Communication Accounting

For a nominal input target $B = 2^b$, coefficient counts follow Appendix M. RoKoKo uses native sets with $N = 2^{22}, 2^{24}, 2^{26}, 2^{28}, 2^{30}$ over an approximately 50-bit field; KoalaBear has about 31 bits, so its achieved

[19] We count source-file lines, including comments and blank lines, in the main workspace at release commit `029cece`; generated SIS tables and third-party dependencies are excluded.

Table 15: Upstream source revisions for the hash-PCS measurements; security profiles are in Table 14.

Implementation	Revision	Implementation	Revision
Plonky2 FRI	`e1c2d354`	Binius64 BaseFold	`6a179536`
Plonky3 FRI and STIR	`3acc8b70`	Flock Ligerito	`b684b125`
WHIR (Plonky3)	`3acc8b70`	WHIR (WorldFnd)	`c03a4a51`
		BaseFold (SP1)	`edc07b68`

capacities are slightly below the nominal 32-bit targets. The opening field need not be the coefficient field. Appendix M specifies the opening-point dimensions, equality-weight workloads, univariate batching, and omitted packing reductions.

The hash harness uses implementation-specific input and point samplers. Its Akita adapter samples uniformly from each full coefficient field and the corresponding extension-field opening domain; WorldFnd WHIR likewise uses uniform Goldilocks coefficients and cubic-extension opening points. Nominal committed bits thus record field capacity rather than sampled entropy; the runs do not constitute a controlled comparison across different fields and opening domains.

Akita's commitment payload is 128 bytes. Its self-describing `CommittedGroup` carries approximately 214–215 additional bytes of public profile and shape metadata, which the refreshed hash results report as excluded verifier context. Hash communication totals include the commitment, the separately transmitted evaluation, and the opening proof. Query points and reusable public parameters are excluded, although the point dimensions still affect the computation. Greyhound's contextual wire encoding excludes public u_1 and the fold schedule, which are verifier context. A dash denotes an unsupported input or an unavailable separate measurement.

N.3 Jolt Measurement Artifact

The measured integration uses Jolt revision `f09bbc1e3`, Akita revision `69438de6`, and `dory-pcs 0.4.2`. The reproduction artifact retains all 72 runs, exact commands, hardware details, timings, and traces. Prover time uses nanosecond root durations; proof size is the complete serialized Jolt proof, rather than only its PCS component. Appendix I describes the integration.

N.4 Detailed Hash-Based PCS Measurements

Below are all measurements from the hash cohorts described above; fields and coefficient counts follow Appendix M.

Table 16: Detailed hash PCS measurements. Verification uses one thread. Communication lists the serialized commitment (Comm.), the opening proof, and the total, which adds the separately transmitted evaluation. RSS includes the dense polynomial; Prep. is reusable preprocessing time and State its persistent storage. Rows marked $^{(1)}$ use batched univariate FRI/STIR once $\log_2 N > 23$.

		Commit (s)		Open (s)		Verify	Communication			RSS (GiB)		Prep.	State
Bits	Scheme	1 thr.	8 thr.	1 thr.	8 thr.	(ms)	Comm. (B)	Proof (KB)	Total (KB)	1 thr.	8 thr.	(s)	(GiB)
2^{27}	Akita (p_{32})	0.0391	0.00827	0.51	0.143	6.0	128	57.7	57.9	0.108	0.147	0.0034	0.0020
	Akita (p_{64})	0.0274	0.00817	0.331	0.0966	3.9	128	59.4	59.6	0.103	0.133	0.0034	0.0024
	Akita (p_{128})	0.0366	0.00977	0.242	0.0789	4.2	128	61.7	61.8	0.118	0.153	0.0059	0.0049
	Plonky2 FRI	3.41	0.548	2.79	1.2	14.5	520	637.2	637.8	0.958	0.957	0.0000	0.0000
	Plonky3 FRI	0.982	0.205	1.15	0.241	7.1	33	410.6	410.6	1.18	1.17	0.0271	*unknown*
	Plonky3 STIR	0.985	0.21	1.63	0.43	5.8	34	290.9	290.9	1.18	1.17	0.0271	*unknown*
	WHIR (Plonky3)	0.0777	0.0205	0.52	0.146	3.4	33	211.4	211.4	0.235	0.24	0.0013	*unknown*
	Binius64 BaseFold	0.0296	0.00664	0.0115	0.00725	0.4	32	308.6	308.7	0.0918	0.0913	0.0031	*unknown*
	Flock Ligerito	0.0541	0.0094	0.0709	0.0185	1.2	4137	408.9	413.0	0.0896	0.0905	0.0001	0.0000
	WHIR (WorldFnd)	0.389	0.0673	2.25	0.348	1.0	56	196.6	196.7	0.174	0.181	0.0002	0.0000
	BaseFold (SP1)	2.13	2.17	3.25	3.25	25.0	32	1179.4	1179.5	1.33	1.33	0.0000	*unknown*

Table 16 (continued)

Bits	Scheme	Commit (s)		Open (s)		Verify	Communication			RSS (GiB)		Prep.	State
		1 thr.	8 thr.	1 thr.	8 thr.	(ms)	Comm. (B)	Proof (KB)	Total (KB)	1 thr.	8 thr.	(s)	(GiB)
2^{29}	Akita (p_{32})	0.141	0.0221	0.703	0.176	7.5	128	58.2	58.4	0.222	0.262	0.0035	0.0020
	Akita offload (p_{32})	0.14	0.0214	1.07	0.225	6.8	128	62.6	62.7	0.232	0.264	0.0091	0.0020
	Akita (p_{64})	0.0857	0.0168	0.423	0.113	4.6	128	61.0	61.1	0.19	0.227	0.0040	0.0029
	Akita offload (p_{64})	0.0842	0.0153	0.826	0.187	4.8	128	67.4	67.5	0.205	0.235	0.0123	0.0039
	Akita (p_{128})	0.113	0.0216	0.43	0.103	5.1	128	62.5	62.6	0.188	0.223	0.0071	0.0059
	Plonky2 FRI	13.8	2.37	11.3	4.89	17.9	520	780.3	780.8	3.82	3.82	0.0000	0.0000
	Plonky3 FRI$^{(1)}$	2.24	0.469	2.31	0.458	7.8	33	448.4	448.5	2.41	2.41	0.0617	*unknown*
	Plonky3 STIR$^{(1)}$	2.24	0.465	3.24	0.861	6.0	34	307.3	307.4	2.41	2.41	0.0623	*unknown*
	WHIR (Plonky3)	0.347	0.0977	2.3	0.715	3.9	33	251.7	251.7	0.928	0.936	0.0055	*unknown*
	Binius64 BaseFold	0.134	0.0406	0.0416	0.0248	0.5	32	390.4	390.4	0.358	0.357	0.0121	*unknown*
	Flock Ligerito	0.211	0.0419	0.338	0.0799	1.3	4137	340.8	345.0	0.364	0.366	0.0002	0.0000
	WHIR (WorldFnd)	1.78	0.31	9.55	1.46	1.1	56	221.6	221.6	0.685	0.723	0.0002	0.0000
	BaseFold (SP1)	2.98	3	3.37	3.27	25.1	32	1182.5	1182.6	1.56	1.56	0.0000	*unknown*
2^{31}	Akita (p_{32})	0.536	0.0748	1.35	0.275	11.7	128	59.5	59.6	0.627	0.663	0.0079	0.0049
	Akita offload (p_{32})	0.534	0.0736	1.85	0.364	10.2	128	62.7	62.9	0.671	0.696	0.0213	0.0039
	Akita (p_{64})	0.303	0.0499	0.815	0.191	7.3	128	62.4	62.5	0.421	0.454	0.0076	0.0059
	Akita offload (p_{64})	0.299	0.0463	1.13	0.24	5.9	128	64.7	64.8	0.487	0.519	0.0307	0.0078
	Akita (p_{128})	0.41	0.0742	0.85	0.188	9.1	128	64.3	64.4	0.507	0.507	0.0138	0.0117
	Plonky2 FRI	55.5	9.5	45.7	20.3	19.5	520	846.0	846.5	15.3	15.3	0.0000	0.0000
	Plonky3 FRI$^{(1)}$	2.61	0.686	2.3	0.46	7.8	33	452.7	452.9	2.78	2.78	0.0607	*unknown*
	Plonky3 STIR$^{(1)}$	2.6	0.687	3.24	0.857	6.1	34	312.4	312.6	2.79	2.78	0.0603	*unknown*
	WHIR (Plonky3)	1.43	0.445	9.34	3.04	4.4	33	279.9	279.9	3.7	3.7	0.0270	*unknown*
	Binius64 BaseFold	0.586	0.23	0.165	0.0923	0.6	32	472.8	472.9	1.42	1.42	0.0498	*unknown*
	Flock Ligerito	0.911	0.165	1.11	0.265	1.2	4137	494.6	498.8	1.35	1.35	0.0002	0.0000
	WHIR (WorldFnd)	7.77	1.37	40.9	6.3	1.2	56	256.9	256.9	2.73	2.85	0.0002	0.0000
	BaseFold (SP1)	5.53	5.39	3.51	3.32	25.4	32	1194.8	1194.9	2.5	2.5	0.0000	*unknown*
2^{33}	Akita (p_{32})	2.08	0.314	2.32	0.449	18.6	128	60.9	61.0	1.21	1.26	0.0159	0.0107
	Akita offload (p_{32})	2.2	0.367	3.33	0.586	11.9	128	63.3	63.4	1.33	1.35	0.0667	0.0156
	Akita (p_{64})	1.25	0.32	1.69	0.333	9.9	128	62.9	63.1	1.33	1.38	0.0181	0.0156
	Akita offload (p_{64})	1.28	0.314	2.47	0.448	7.5	128	65.0	65.1	1.5	1.54	0.0781	0.0156
	Akita (p_{128})	1.64	0.375	1.98	0.348	12.2	128	64.8	65.0	1.41	1.44	0.0340	0.0313
	Akita offload (p_{128})	1.66	0.37	2.47	0.441	8.7	128	67.1	67.3	1.5	1.54	0.0991	0.0313
	Plonky2 FRI	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*
	Plonky3 FRI$^{(1)}$	5.16	1.64	2.14	0.462	7.9	33	470.3	470.9	4.28	4.28	0.0595	*unknown*
	Plonky3 STIR$^{(1)}$	5.16	1.65	3.1	0.849	6.2	34	329.9	330.6	4.29	4.28	0.0602	*unknown*
	WHIR (Plonky3)	4.72	1.54	36.9	12.4	4.7	33	305.8	305.8	13.7	13.7	0.0616	*unknown*
	Binius64 BaseFold	2.5	1.16	0.6	0.34	0.7	32	569.4	569.5	5.67	5.67	0.1971	*unknown*
	Flock Ligerito	3.47	0.763	4.47	1.06	1.6	4137	524.8	529.0	5.38	5.41	0.0002	0.0000
	WHIR (WorldFnd)	34.6	5.91	178	27.4	1.3	56	281.5	281.5	10.9	11	0.0002	0.0000
	BaseFold (SP1)	18.8	18.8	3.87	3.71	26.4	32	1244.0	1244.0	6.25	6.25	0.0000	*unknown*
2^{35}	Akita (p_{32})	8.43	1.24	5.7	0.946	32.2	128	61.0	61.1	4.37	4.4	0.0301	0.0215
	Akita offload (p_{32})	10.7	1.75	8.33	1.33	14.1	128	63.7	63.9	4.47	4.5	0.1432	0.0313
	Akita (p_{64})	4.62	1.3	4.89	0.823	18.0	128	63.0	63.2	4.51	4.61	0.0359	0.0313
	Akita offload (p_{64})	4.59	1.27	5.46	0.997	10.1	128	66.8	67.0	4.7	4.77	0.1356	0.0313
	Akita (p_{128})	5.66	1.43	6.35	0.951	22.2	128	65.0	65.2	4.75	4.81	0.0653	0.0625
	Akita offload (p_{128})	5.74	1.41	6.21	1.01	10.7	128	67.9	68.1	4.91	4.98	0.2066	0.0625
	Plonky2 FRI	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*
	Plonky3 FRI$^{(1)}$	17.2	6.27	2.91	0.687	8.5	33	532.7	535.3	12.2	12.2	0.0598	*unknown*
	Plonky3 STIR$^{(1)}$	17.1	6.27	3.84	1.09	6.9	34	401.6	404.2	12.2	12.2	0.0620	*unknown*
	WHIR (Plonky3)	15.5	5.23	39.1	12.7	6.1	33	410.4	410.4	18.9	18.9	0.0624	*unknown*
	Binius64 BaseFold	10.7	5.43	2.34	1.31	0.9	32	666.7	666.8	22.7	22.7	0.7512	*unknown*
	Flock Ligerito	15.3	3.02	18.6	4.46	1.5	4137	570.5	574.6	21.5	21.6	0.0002	0.0000
	WHIR (WorldFnd)	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*
	BaseFold (SP1)	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*