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, $\widetilde{O}_{K,\lambda}(N)$ 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 . . . 15
- 2.3 Optimized Digit Range Check . . . 17
- 2.4 Optimizing the Full Recursion . . . 18
- 2.5 Two Opening Representations . . . 19
- 2.6 Checking Ring Relations with or without Quotients . . . 22
- 2.7 Commitment Compression . . . 23
- 2.8 Bounding Response Coefficients and the Total Squared Norm . . . 24
- 2.9 Batching and Distributed Proving . . . 26
- 2.10 Offline Parameter Selection and Independent Validation . . . 27

3 Preliminaries . . . 27
- 3.1 Cyclotomic Rings, Extension Fields, and Challenge Subrings . . . 27
- 3.2 Sparse and Filtered Folding Challenges . . . 29
- 3.3 Gadget Decomposition and Module-SIS . . . 31
- 3.4 Multilinear Extensions and Sum-Check . . . 31
- 3.5 Equality-Factored Sum-Check . . . 33
- 3.6 Quotient-Lift Ring Switching . . . 33
- 3.7 Tensor Reduction to the Extension Field . . . 34
- 3.8 Polynomial Commitments and Coordinate-Wise Special Soundness . . . 35

4 Setup and Commitments . . . 37
- 4.1 Setting . . . 37
- 4.2 Schedule Topology and Fold Configuration . . . 39
- 4.3 Setup . . . 40
- 4.4 Per-Matrix Ring Dimensions and Challenge Families . . . 43
- 4.5 Commitment . . . 46
- 4.6 Commitments Formed before the Fold . . . 51

5 One Fold for an Opening Batch . . . 51
- 5.1 Inputs and Parameters . . . 52
- 5.2 Shared Tensor Reduction . . . 54
- 5.3 Forming the Partial Opening Evaluations . . . 55
- 5.4 Binding the Partials and Folding the Sources . . . 57
- 5.5 The Witness Passed to the Next Level . . . 59
- 5.6 Interaction and Special Cases . . . 61

6 Norm Checks for the Recursive Witness . . . 63
- 6.1 Digit Range Check . . . 63
- 6.2 Certifying the ℓ_2 Response Norm . . . 65

7 Checking the Ring Relations . . . 68
- 7.1 Opening-Method Evaluation Rows . . . 68
- 7.2 The Common Ring Relation . . . 70
- 7.3 Reducing Ring Equations to Field Identities . . . 70
- 7.4 The Fused Sum-Check . . . 76
- 7.5 Verifier Evaluation of the Batched Row . . . 77

8 Recursive Opening and Terminal Handling . . . 79
- 8.1 Composing the Folds . . . 79
- 8.2 Terminal Levels (Tail Handling) . . . 79

9 Verifier Offloading . . . 81

9.1 The Setup Claim and Setup-Product Sum-Check . . . 81
9.2 Passing the Setup Opening to the Next Fold . . . 82
9.3 Full Schedules and Admissibility . . . 83
9.4 The Constant-Root Verifier . . . 85
9.5 The Assembled Protocol . . . 88
10 Security of Akita . . . 89
10.1 Completeness . . . 92
10.2 Knowledge Soundness . . . 96
10.3 Schedules Chosen from a Public Family . . . 125
11 Response Chunking for Distributed Proving . . . 127
11.1 The Distributed-Response Obstruction . . . 127
11.2 The Response-Chunked Witness . . . 127
11.3 One Relation over Chunked Witness Columns . . . 128
11.4 Local Images and the Recursive Commitment . . . 129
11.5 Distributed Sum-Check and Verifier Work . . . 130
11.6 Cost, Completeness, and Security . . . 131
12 Offline Planner and Parameter Selection . . . 133
12.1 Parameter-Selection Approaches . . . 134
12.2 Choosing a Complete Schedule . . . 134
12.3 Predicting the Next Fold . . . 135
12.4 Searching Across the Recursion . . . 135
12.5 Validating and Using Schedules . . . 135
13 Implementation and Evaluation . . . 135
13.1 Implementation and Experimental Methodology . . . 136
13.2 Comparison with Lattice-Based PCSs . . . 136
13.3 Comparison with Hash-Based PCSs . . . 138
13.4 Jolt End-to-End . . . 141
14 Conclusion and Future Work . . . 142
A Multilinear Evaluation from Structured Factors . . . 147
A.1 From Tensor Factors to Vector Addresses . . . 147
A.2 The Common Evaluation Problem . . . 148
A.3 Finite-State Contraction . . . 149
A.4 Affine Addresses on Intervals . . . 150
A.5 Sharing Equality Work and Combining Families . . . 152
B Evaluating Akita's Public Weights . . . 154
B.1 Quotient-Free Relation Weights . . . 154
B.2 Opening, Range, and Norm Weights . . . 156
B.3 Factored Blocks in the Relation . . . 157
B.4 Shared Setup Views and Offloaded Weights . . . 158
C SIS Security Estimation . . . 159
D Certifying Operator-Norm Challenge Filters . . . 164
D.1 A Deterministic Fixed-Point Predicate . . . 164
D.2 An Exact Accepted-Support Certificate . . . 165
D.3 Concrete Certificates . . . 166
E Akita in Jolt . . . 166
E.1 Committed Polynomials . . . 167
E.2 From Column Claims to Commitment Openings . . . 167
E.3 One Heterogeneous Akita Proof . . . 168
E.4 Schedules and Setup Offloading . . . 168
E.5 Transcript Grinding in the Backend . . . 169
F Corrections and Security Mismatches in Prior Work . . . 169
F.1 Corrections to Protocol Descriptions . . . 169
F.2 Released Implementations . . . 170
G Response Bounds and Parameter Estimates . . . 174
G.1 Bounds for Honest Responses . . . 174
G.2 From Response Bounds to Completeness . . . 177
G.3 A Response Model for Parameter Selection . . . 179
G.4 A Certified Alternative for Digit Squared Norms . . . 183
H PCS Benchmark Scope and Parameters . . . 183

I Benchmark Methodology and Detailed Measurements 185
I.1 Software and Measurement Procedure 186
I.2 Inputs and Communication Accounting 187
I.3 Jolt Measurement Artifact 187
I.4 Detailed Hash-Based PCS Measurements 188

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 [48,44,28] 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) [58]. 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 [47], BaseFold [92], and Ligerito [84]; and the binary-field line of Binius [37,38], Blaze [27], and Bolt [50]. These now underpin deployed proof systems at scale, including Succinct's SP1 [90], Lighter's Plonky2-based exchange rollup [72], and the many provers in the Ethereum Foundation's real-time proving effort, Ethproofs [39,40].

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 [7,41]) 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 $160KB$ for non-trivial instances [9,70]. 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 [81,79]. A flurry of recent constructions [18,81,79,55,59,60,61,56,54] 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 [89] 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 [71], 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 [81] and Hachi [79] obtain small proofs from standard Module-SIS,[4] but verify in time square-root in the polynomial size. Alternate designs such as RoKoKo [61] and Grand Danois [56] 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 [47,50] 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 [2,63], an extensive history of cryptanalysis, and underlying the security of NIST-standardized ML-DSA [78].

by leaning on stronger, more structured assumption variants (vanishing SIS [33]) 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 [25,3,34], 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 [81,79]. 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 [86]. Appendix F 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 C. 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 F for qualifications.

PCS	Binding assumption	Proof size	Verifier	Artifact
Greyhound [81]	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 [79]	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 [54]	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 [61]	vSIS	$\widetilde{O}_\lambda(\log N)$; 115 KB for $N = 2^{26}$–2^{30} (Table 8 in Section 13.2)	$\widetilde{O}_\lambda(\log N)$	End-to-end Rust
Grand Danois [56]	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 8 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 Section 2.4 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 [49], 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 [46] 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. [36].

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 [38] to connect the original base-field evaluation to the packed claim. This reduction supplies the binding missing from Hachi's base-field shortcut [79, Section 3.2, p. 14] (Appendix F.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 [53], 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 [56] 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 F.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 [81, §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, we provide an option for Akita to be able to prove across

[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 3-bit balanced digits, or equivalently the set $\{-4, -3, -2, -1, 0, 1, 2, 3\}$.

multiple machines, keeping low (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]. Its efficiency is around 3 orders of magnitude over 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 [74]. Lyubashevsky and Seiler [76] 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 [81]. 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 [79]. Its trace identity for embedding extension-field inner products into a cyclotomic ring generalizes the trace technique of LNP22 [75]; its reduction of ring relations to field sum-check is the quotient-based ring-switching method of Huang, Mao, and Zhang [53]. 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 [19,75]. 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 still rests on standard Module-SIS.

Lattice PCSs with single-level folding. CELPC [55] obtains square-root proof size and verification with a transparent setup. Jindo [54] 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 [42] 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 [34] 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 [42,3,34]. 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 [59] develops verifier-succinct reductions of knowledge for exact norm bounds; RoK and Roll [60] adds structured random projections; SALSAA [62] integrates sum-check to obtain linear prover time; and RoKoKo [61] 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 [56] 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 [82,83] obtained pay-per-bit commitment costs over small fields; Cyclo [45] 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 [85] 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. We compare this construction with Akita's parallel development below. 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 [61, 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 D. 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 [92], Ligerito [84], and WHIR [9] to Binius [37,38], 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 $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N^{1/\kappa_{\mathsf{root}}})$ (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 F 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. Akita has a long technical development, but the paper is modular, and we do not expect every reader to follow it linearly. We recommend starting with Section 2, which 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 E.

Building Akita. The overview presents the design ideas in order of their role in the result. The main body gives the complete specification in dependency order. We begin in Section 3 with the algebraic tools, fix the setup and commitments in Section 4, and construct one fold for an opening batch in Section 5. Sections 6 and 7 then establish its norm and algebraic checks, and Section 8 closes the recursion and handles the terminal level. This is the main route for seeing how the pieces of one recursive opening fit together.

From one fold to the full system. With the fold and its composition rules in place, Section 9 uses a second input group to authenticate the setup computation recursively. Section 10 then proves completeness and knowledge soundness for the assembled protocol. We next turn to how the protocol is run: Section 11 divides its work among machines, while Section 12 selects and validates the complete schedule.

Experiments and supporting material. Section 13 reports the implementation and benchmarks, and Section 14 closes the main body. Readers looking for supporting machinery can use the appendices as reference: Appendix A develops the structured-tensor evaluator, Appendix C documents SIS estimation, Appendix D certifies the operator-norm filters, Appendix E specifies the Jolt interface, and Appendix F records the corrections needed for security-matched comparisons with prior work.

2 Technical Overview

We start from the Hachi fold [79] 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 wants to prove $\widetilde{f}(\mathbf{r}) = v$, where $\mathbf{r} \in E^\ell$ for an extension field $E/\mathbb{F}_q$.

Our goal is to reduce this claim to an opening of a new multilinear polynomial with fewer Boolean evaluations, while preserving its connection to the original commitment and claimed value. We start by dividing the Boolean evaluations into B equal-length blocks and retaining a random linear combination of the blocks. We will specify the block length after introducing the ring encoding below. 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. The resulting response would give no useful Module-SIS bound.

Lattice identification and zero-knowledge protocols address this problem with challenges drawn from a large family of short elements [74,76]. Greyhound and Hachi use this idea in the cyclotomic ring

$$R_q := \mathbb{F}_q[X]/(X^d + 1).$$

The challenge family $\mathcal{C} \subset R_q$ has enough elements for soundness, small coefficient norms to control response growth, and invertible differences between distinct challenges to support extraction. Write $\mathbf{c} = (c_i)_{i\in[B]}$ for the challenges. For the short vector $\mathbf{s}_i$ representing block i, the response is

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

Ring multiplication computes a negacyclic convolution. Its coefficient one-norm controls how much a challenge can enlarge the response; the exact challenge conditions and bounds appear in Section 3.2.

This linear combination is only the start of an opening proof. We must connect the blocks to their commitment, connect their partial opening 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 uses the ring-switching technique of Huang, Mao, and Zhang to check it by field sum-check [18,81,79,53]. We explain these ingredients in the order in which an opening uses them.

Representing the evaluation in the ring. For the warm-up, start with an extension-field opening supplied by the standard tensor reduction [38, Theorem 3.5] (Section 3.7); we reuse $\mathbf{r}$ and v for its point and claimed value. We now connect this field evaluation to the commitment ring R_q.

The ring encoding must let us recover field inner products from ring products. Hachi's trace construction provides a packing map $\psi : E^{d/k} \to R_q$, where $k = [E : \mathbb{F}_q]$, with this property. The field occupies a specified subfield of R_q; arbitrary placement in coefficient slots would not preserve multiplication by extension-field values. Section 3.1 gives the embedding, its dimension conditions, and the trace identity, and Section 7.1

gives the resulting opening row. Here we use the public linear functional $\mathcal{T}_{\mathbf{r}_{\mathsf{pack}}} : R_q \to E$ supplied by that identity.

After packing, write the blocks as $\mathbf{f}_i \in R_q^M$, so the packed vector contains $N_R := BM$ ring elements. Let $\mathbf{a} \in R_q^M$ contain the within-block opening weights. A partial opening evaluation in the ring records each block's contribution:

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

Here $\chi_{\mathsf{blk}}(i) = \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{blk}}, i)$ is the weight of block i; the remaining packed-coordinate weights enter $\mathcal{T}$. Hachi sends v_{R} and checks its trace explicitly. The partials $(e_i)_i$ now carry the evaluation obligation, but revealing them all would give back much of the saving. We need to bind them with a short public message instead.

Committing through small digits. We first explain the short block representations $\mathbf{s}_i$. For an even base b, write each coefficient using digits in $\mathcal{A}_b = \{-b/2, \ldots, b/2 - 1\}$. Let $\mathbf{G}_{b,n}^{-1}$ be a fixed decomposition of n ring elements, and let $\mathbf{G}_{b,n}$ recompose those digits with powers of b:

$$\mathbf{G}_{b,n} \mathbf{G}_{b,n}^{-1}(\mathbf{x}) = \mathbf{x} \quad \text{in } R_q. \tag{3}$$

Thus the digits give a short representation even when $\mathbf{x}$ has arbitrary coefficients modulo q. The exact depths, matrix dimensions, and decomposition convention are specified in 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:

$$\mathbf{t}_i := \mathbf{A}\mathbf{s}_i, \qquad \hat{\mathbf{t}}_i := \mathbf{G}_{b_1,n_A}^{-1}(\mathbf{t}_i). \tag{4}$$

Sending all B inner images would cost too much. An outer matrix $\mathbf{B}$ instead commits to their concatenated digits:

$$\mathbf{u} := \mathbf{B}\hat{\mathbf{t}}, \qquad \hat{\mathbf{t}} := (\hat{\mathbf{t}}_i)_{i \in [B]}. \tag{5}$$

The public commitment is $\mathbf{u}$. The first decomposition supplies short openings for the input blocks; the second lets one short message bind the list of their inner images. Section 4.5 specifies Akita's commitment construction and its matrix shapes.

Binding the partial opening evaluations before folding. We use the same pattern for the partial opening evaluations. Decompose e_i as $\hat{\mathbf{e}}_i = \mathbf{G}_{b_1,1}^{-1}(e_i)$ and publish

$$\mathbf{v} := \mathbf{D}\hat{\mathbf{e}}, \qquad \hat{\mathbf{e}} := (\hat{\mathbf{e}}_i)_{i \in [B]}. \tag{6}$$

The matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{D}$ are independent in this baseline. The input commitment $\mathbf{u}$ is fixed before the opening point is known. After receiving the point, the prover sends $\mathbf{v}$ and v_{R}; only then does the verifier sample $(c_i)_i$ and the prover form $\mathbf{z}$. Both the inner images and the evaluation partials are therefore fixed before the coefficients used to compare them are known. Akita's fold transcript is specified in Section 5.

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

$$\mathbf{A}\mathbf{z} = \sum_{i \in [B]} c_i \mathbf{G}_{b_1,n_A} \hat{\mathbf{t}}_i, \tag{7}$$

$$\mathbf{a}^{\top} \mathbf{G}_{b,M} \mathbf{z} = \sum_{i \in [B]} c_i \mathbf{G}_{b_1,1} \hat{\mathbf{e}}_i. \tag{8}$$

The first ties the response to the committed blocks; the second ties that same response to their partial evaluations. Opening $\mathbf{u}$ and $\mathbf{v}$ and reconstructing v_{R} supplies the other three constraints: $\mathbf{u} = \mathbf{B}\hat{\mathbf{t}}$, $\mathbf{v} = \mathbf{D}\hat{\mathbf{e}}$, and $v_{\mathsf{R}} = \sum_i \chi_{\mathsf{blk}}(i) e_i$.

A response coefficient can be larger than an input digit, so the response needs its own decomposition. If the input decomposition uses δ digits per coefficient, then $\mathbf{z}$ has $M\delta$ ring coordinates. At response base b_z and bound-selected depth τ, its recomposition map is

$$\mathbf{G}_{\mathbf{z}} := \mathbf{I}_{M\delta} \otimes (1, b_z, \ldots, b_z^{\tau-1}), \qquad \mathbf{z} = \mathbf{G}_{\mathbf{z}} \hat{\mathbf{z}}.$$

Section 5.5 specifies how the response bound determines this depth. Unlike a full-field decomposition, this depth need only cover the admitted response interval. The private relation witness is $\mathbf{w}_0 = (\hat{\mathbf{z}}, \hat{\mathbf{e}}, \hat{\mathbf{t}})$. Section 7.2 specifies the complete Akita relation, including its opening-method and compression choices.

Certifying the digit bounds. The equations establish consistency modulo q and $X^d + 1$; they do not establish that the supplied digits are small. We also prove membership in the prescribed alphabet $\mathcal{A}$, using

$$Q_{\mathcal{A}}(w) := \prod_{a \in \mathcal{A}} (w - a), \qquad Q_{\mathcal{A}}(w) = 0. \tag{9}$$

Sum-check proves this identity for every entry of the digit vector. Recomposition and the range check serve different purposes: one connects the digits to the intended values, while the other bounds their size. Together they certify the short witness needed by the next fold. This is the norm reset that prevents response growth from accumulating indefinitely. Akita's range-check construction appears in Section 6.1. This check concerns the successor witness; the source recovered by extraction requires a separate argument below.

Returning to a smaller opening claim. The relation lives over R_q, whereas sum-check runs over a field. Substituting a random $X = \alpha$ would not preserve reduction modulo $X^d + 1$. Hachi uses the HMZ ring switch [53]: a private quotient records the multiples of $X^d + 1$ discarded by each ring equation, making it a polynomial identity that we can evaluate at α. We compare this route with Akita's quotient-free alternative in Section 2.6.

The quotient digits join the response and auxiliary digits in

$$\mathbf{w}_{\mathsf{next}} := (\hat{\mathbf{z}}, \hat{\mathbf{e}}, \hat{\mathbf{t}}, \hat{\mathbf{r}}).$$

The range and relation checks leave one opening of the multilinear polynomial with these Boolean evaluations, padded to a power-of-two length. We have returned to the original type of claim. Up to security and digit factors, the response has length M and its auxiliaries have length B. Taking $B \approx M \approx \sqrt{N_R}$ therefore gives a square-root reduction. Packing changes N to N_R only by fixed-security factors, which we suppress in the asymptotic costs below.

Binding from weak openings. Consider two accepting executions that differ in one challenge c_i. The recursively authenticated digits either yield a short **B** or **D** collision, or fix the same auxiliaries in both executions. Write Δc_i and $\Delta \mathbf{z}$ for the challenge and response differences. The challenge family makes Δc_i a unit, so in the latter case subtraction recovers

$$\mathbf{s}_i = (\Delta c_i)^{-1} \Delta \mathbf{z}, \qquad \Delta c_i \mathbf{s}_i = \Delta \mathbf{z}. \tag{10}$$

The source satisfies the commitment and evaluation equations, but an inverse of a short ring element need not be short. Extraction instead bounds 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, supplies binding [13, Definition 4.2 and Lemma 4.3][81, Lemma 2.11][79, Lemma 7]. If distinct sources $\mathbf{s}_i, \mathbf{s}'_i$ have the same inner image and bounded products with short units ρ_i, ρ'_i, then

$$\mathbf{z}_A := \rho'_i(\rho_i \mathbf{s}_i) - \rho_i(\rho'_i \mathbf{s}'_i) = \rho_i \rho'_i(\mathbf{s}_i - \mathbf{s}'_i) \tag{11}$$

is a nonzero **A** kernel vector. It is short because each term is a short multiplier times a bounded product. Thus **A** is sized for this collision, whereas **B** and **D** bind digit differences directly. Starting at the directly checked terminal, we apply extraction backward to the root. Lemma 10.23 and Theorem 10.40 give the full argument, including the source and padding conventions.

Allowing different ring dimensions. A common ring makes this baseline easy to describe. Akita need not use the same ring dimension for every relation. The **A** ring governs the folded response, while the **B** and **D** rings serve the outer and opening commitments. We choose d_A, d_B, d_D for those separate tasks, then reduce each ring equation to a field identity before batching them. The same principle applies to the compression maps introduced later. Remark 7.2 and Section 7.3 formalize this freedom. It lets each relation use the ring dimension it needs without using the largest dimension for every commitment.

2.2 Setup Offloading and the Constant-Root Verifier

In the warm-up, we used Hachi's fold to reduce an opening to a smaller one, but the verifier still had to process both the folding challenges and the public commitment matrices. Fewer blocks mean fewer challenges, yet also longer blocks and a larger matrix computation. To improve verification, we need to remove this second cost without restoring it elsewhere in the verifier's work.

We do so by proving the setup-dependent computation and passing its remaining opening claim to the next fold. This is our setup-offloading technique. It lets us obtain verifier time $\widetilde{O}(N^{1/\kappa_{\mathsf{root}}})$ for any fixed $\kappa_{\mathsf{root}} \geq 2$, with the same asymptotic prover time and proof size as the baseline and the same constant-root bound on the expanded verifier setup. We first identify the computation to offload, then explain how its opening is checked and derive the resulting recursion.

The square-root verification cost. Let us return to the final point $\mathbf{r}_2$ of the fused relation sum-check. We have reduced the witness obligation to a new opening, but must still evaluate the public relation polynomial m_{τ_1} at that point. Here τ_1 fixes the combination of relation rows. Write the resulting value as

$$m_{\tau_1}(\mathbf{r}_2) = m_{\mathsf{local}}(\mathbf{r}_2) + \sigma_S, \tag{12}$$

where σ_S is the contribution from the $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{D}$ setup coefficients. We evaluate the remaining gadget, challenge, equality, quotient, and compression contributions directly. A short seed describes the setup coefficients, but does not by itself save the work of generating and processing them.

To see the tradeoff, divide the current witness into B blocks of length M, with $BM \approx N$ after suppressing the fixed representation factors from our warm-up. Most local weights can be evaluated from their separate factors, using the methods in Appendix B. The verifier must nevertheless derive and evaluate all B folding challenges $(c_i(\alpha))_{i\in[B]}$. This leaves $\widetilde{O}_\lambda(B)$ work even if we remove the setup computation.

We also have to account for the setup rows. The active $\mathbf{A}$ view uses one folded block and has width proportional to M. The $\mathbf{B}$ and $\mathbf{D}$ views use block-indexed auxiliaries and have width proportional to B. Thus the direct verifier's accounting is

$$\begin{aligned} \text{challenge-dependent rows}: &\quad \widetilde{O}_\lambda(B), \\ \text{public setup rows}: &\quad \widetilde{O}_\lambda(M+B), \\ \text{total verifier work}: &\quad \widetilde{O}_\lambda(B+M) = \widetilde{O}_\lambda\left(B + \frac{N}{B}\right). \end{aligned} \tag{13}$$

At the balanced point $B \approx M \approx \sqrt{N}$, both terms are square-root. Choosing fewer blocks reduces the challenge work but enlarges the $\mathbf{A}$ scan, so a naive wide fold makes the verifier slower.

We mark these costs on the relation in Figure 1. Amber blocks contain the B challenge evaluations, blue blocks contain the setup coefficients that we offload, and green blocks retain short factored descriptions that the verifier evaluates directly.

Offloading the matrix evaluations. We ask the prover to supply σ_S, then prove that it is correct. To combine the matrix contributions, we expand one flat setup sequence $(S_0, S_1, \ldots)$ from the public seed and let every matrix read a prefix under its own shape. This shared-prefix layout also appears in optimized Greyhound implementations [81]. Whenever several views use the same coefficient, we add their weights. Their total contribution is then one inner product:

$$\sigma_S = \sum_{p < N_{\mathsf{active}}} S_p \, \omega_{\mathsf{setup}}(p). \tag{14}$$

Here N_{active} is the longest active prefix, and $\omega_{\mathsf{setup}}(p)$ combines the public weights of all uses of S_p. These weights depend on the matrix layouts, relation challenges, and sum-check point. Appendix B.4 gives their exact formulas. Sharing coefficients is covered by the reduction of Lemma 10.19: extraction identifies one offending matrix view and reduces its binding failure to the ordinary Module-SIS assumption.

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: $\widetilde{O}_\lambda(M+B)$

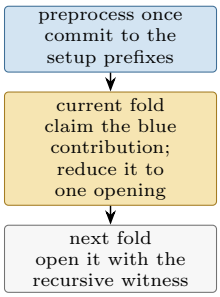

Fig. 1: Verifier cost and setup offloading. Left: after substituting $X = \alpha$, factored blocks (green) cost $\widetilde{O}_\lambda(\log N)$, challenge blocks (amber) cost $\widetilde{O}_\lambda(B)$, and setup blocks (blue) require the $\widetilde{O}_\lambda(M+B)$ matrix scan. Row batching does not 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 setup-product sum-check. Once the relation sum-check has fixed $\mathbf{r}_2$, we can check the claimed σ_S with a degree-two sum-check. Pad the active setup prefix and its weights with zeros to length 2^{ν_S}, and let $\widetilde{\mathbf{S}^\flat}$ and $\widetilde{\omega}_{\mathsf{setup}}$ be the multilinear polynomials with those Boolean evaluations. The identity is

$$\sigma_S = \sum_{p \in \{0,1\}^{\nu_S}} \widetilde{\mathbf{S}^\flat}(p)\, \widetilde{\omega}_{\mathsf{setup}}(p). \tag{15}$$

We arrive at a claimed value $s_{\rho_S} = \widetilde{\mathbf{S}^\flat}(\rho_S)$ multiplied by $\widetilde{\omega}_{\mathsf{setup}}(\rho_S)$. The verifier computes the second factor from its public description, without enumerating its Boolean evaluations (Appendix B.4). Computing the first factor directly would restore the scan we wanted to avoid. We therefore commit during preprocessing to every setup prefix that a scheduled fold may offload, and authenticate this value through a later opening. Fold h carries the resulting setup opening beside its recursive-witness opening:

$$\mathcal{O}_{h+1} = (\mathcal{S}_h, \mathcal{W}_{h+1}), \qquad \mathcal{S}_h = (C_{S,h}, \rho_S, s_{\rho_S}). \tag{16}$$

We check both groups in the successor through the multi-instance relation of Section 5. A fold may check the incoming setup opening while producing a new one for its successor, so at most one setup opening is pending at any recursive boundary.

We pay for this saving in the next fold: the successor must represent and certify the setup group like any other opening. We offload only while this cost is smaller than the removed scan. The final nonterminal fold must check any pending setup group because the terminal cannot check a new one (Section 9.2).

The constant-root recursion. We can now choose wide folds without paying for their long setup views. Fix a constant $\kappa_{\mathsf{root}} \geq 2$ and set the target scale

$$N_\star := \left\lceil N^{1/\kappa_{\mathsf{root}}} \right\rceil. \tag{17}$$

During each of the first $\kappa_{\mathsf{root}} - 1$ folds, we use $B_j = \widetilde{\Theta}_{\kappa_{\mathsf{root}},\lambda}(N_\star)$ input blocks and block length $M_j = \lceil N_j / B_j \rceil$. The verifier still processes the B_j challenge evaluations, but offloading removes the longer M_j setup scan. The direct $\mathbf{F}/\mathbf{H}$ evaluations remain. Under the regularity conditions of Section 9.4, these and the other local computations fit within $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N_\star)$ work at each early fold.

With this choice, we pass forward a folded response of length about $N_j/N_\star$, $N_\star$-scale auxiliaries, and the carried setup group. The complete successor therefore satisfies

$$N_{j+1} = \widetilde{O}_{\kappa_{\mathsf{root}},\lambda}\left(\frac{N_j}{N_\star} + N_\star\right). \tag{18}$$

After $\kappa_{\mathsf{root}} - 1$ such folds, the remaining claim has size $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N_\star)$; the recursion then continues to the terminal boundary.

For every fixed κ_{root}, we obtain the following costs under these regularity conditions:

$$\begin{aligned} \text{prover time} &= \widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N), \\ \text{proof size} &= \widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(\log N), \\ \text{verifier time} &= \widetilde{O}_{\kappa_{\mathsf{root}},\lambda}\left(N^{1/\kappa_{\mathsf{root}}}\right), \\ \text{expanded } \mathbf{A}/\mathbf{B}/\mathbf{D} \text{ verifier setup} &= \widetilde{O}_{\kappa_{\mathsf{root}},\lambda}\left(N^{1/\kappa_{\mathsf{root}}}\right). \end{aligned} \tag{19}$$

The case $\kappa_{\mathsf{root}} = 2$ recovers the square-root point, while larger fixed κ_{root} gives an arbitrarily small constant root. We retain quasilinear prover work because the witness sizes at successive levels form a geometric series. A fixed number of wide folds preserves logarithmic proof size. The first direct setup scan occurs only after the witness reaches the $N_\star$ scale, giving the same bound for the expanded setup. The regularity conditions and full accounting appear in Sections 9 and 9.4. For concrete parameters, we compare this extra recursive work with the scan it removes and stop offloading once a direct scan is cheaper; the planner of Section 12 makes this choice.

2.3 Optimized Digit Range Check

In the warm-up, we saw why each fold must certify its new digits. We now make that check less expensive. A larger base b shortens the digit decomposition, but the ordinary range polynomial

$$Q_b(w) := \prod_{a \in \mathcal{A}_b} (w - a), \qquad \mathcal{A}_b := \{-b/2, \ldots, b/2 - 1\},$$

has degree b. Checking it directly makes the sum-check messages wider and requires more prover arithmetic at every fold. We want to retain the shorter decomposition without paying this full cost.

A standard GKR reduction organizes the b factors into a product tree and checks each multiplication layer by sum-check [46]. This reduces the degree of each check, but introduces more sequential stages and intermediate evaluation claims. Before building the tree, we can do better by using the particular roots of our range polynomial.

Pairing the allowed digits. We pair each digit k with $-(k+1)$, for $0 \le k < b/2$. Their two factors satisfy

$$(w - k)(w + k + 1) = w(w + 1) - k(k + 1).$$

Thus both allowed digits give the same value of $w(w+1)$. We can check membership among half as many transformed values:

$$Q_b(w) = Q_{\mathsf{sq}}\big(w(w+1)\big), \qquad Q_{\mathsf{sq}}(s) := \prod_{k=0}^{b/2-1} \big(s - k(k+1)\big). \tag{20}$$

This symmetry is one reason we use the balanced alphabet $\{-b/2, \ldots, b/2-1\}$. The new predicate has degree $b/2$ in s; we still have to connect s to the digits already committed by the prover.

Authenticating the transformed polynomial. Let $\widetilde{\mathbf{w}}$ be the multilinear polynomial whose Boolean evaluations are the committed witness digits. We define $\widetilde{\mathbf{s}}$ to be the unique multilinear polynomial satisfying

$$\widetilde{\mathbf{s}}(x) = \widetilde{\mathbf{w}}(x)\big(\widetilde{\mathbf{w}}(x) + 1\big) \qquad \text{for } x \in \{0,1\}^{\mu'}.$$

Our range argument checks Q_{sq} on these Boolean evaluations and leaves one claim $s_{\mathsf{claim}} = \widetilde{\mathbf{s}}(\mathbf{r}_{\mathsf{virt}})$. We do not commit to this polynomial separately. Instead, we connect that claim to the witness through the subsequent relation sum-check:

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

The restriction to Boolean inputs matters. At a general point $\mathbf{r}$, $\widetilde{\mathbf{s}}(\mathbf{r})$ need not equal $\widetilde{\mathbf{w}}(\mathbf{r})(\widetilde{\mathbf{w}}(\mathbf{r})+1)$. The sum above evaluates the correct multilinear polynomial and lets us use the transformed predicate without adding another commitment. Section 6.1 gives the range identity at a random point and its connection to the relation sum-check.

Reducing the remaining proof and prover work. For larger bases, we build a shallow product tree for the reduced predicate, combining four factors per layer where possible. Compared with a binary tree, this uses slightly wider round polynomials but fewer sequential stages and fewer intermediate claims. Section 6.1 specifies the tree for each supported base and the challenges connecting its layers.

We also use two complementary sum-check optimizations. At range-tree stages without a fused norm term, Gruen's message format removes the common public equality factor from the prover's round polynomial, reducing the degree that must be transmitted (Section 3.5; [49]). The norm-fused final stage uses the standard message format (Table 6). Following the small-value techniques of Dao et al. [36], we reuse computations for repeated combinations of the transformed digits: initially, $w(w+1)$ takes only $b/2$ possible values. The detailed prover algorithm, including these precomputed contributions, appears in the Akita Book [64].

We can now consider larger digit bases without paying the full degree-b range-check cost. The choice still affects the next fold: a larger base reduces the number of digits but allows larger digit differences in the binding analysis. We therefore choose the base together with the rest of the recursion, which brings us to the next part of our construction.

2.4 Optimizing the Full Recursion

We have now reduced the verifier's matrix work and the cost of certifying new digits. To understand how these savings interact, let us follow an opening through the complete recursion. Every object passed to the next witness must be committed, range checked, and folded again. A choice that helps the current fold may therefore leave more work for its successors. We choose the complete schedule with verifier efficiency as our first priority, followed by proof size and prover work (Section 12).

As in Hachi's asymptotic analysis, we repeat the fold until direct checking becomes cheaper. The choices that help us reach that point change as the witness shrinks. We distinguish three cost regimes below.

A running example. To keep the changing dimensions visible, we follow one illustrative schedule throughout the rest of this overview. The input is a single dense polynomial represented by 2^{30} evaluations on the Boolean hypercube, with arbitrary values in the 32-bit base field $K = \mathbb{F}_q$, and one claimed evaluation in the degree-4 extension E/K. Its evaluation array occupies 4 GiB when stored as 32-bit words. This is the same field regime and source scale used in Hachi's concrete discussion [79].

Our illustrative schedule reduces this opening through six committed folds, which we label 0 through 5, followed by a clear terminal level T. We use this example to make the comparisons below concrete; its particular parameters are not requirements of the protocol. Each subsection introduces the entries needed to explain its choice. Section 12 explains how we generate and validate complete schedules.

Early folds. We begin with the largest witnesses, where the first folds dominate prover and verifier time. Aggressive folding reduces the witness quickly. In our running schedule, we offload the root's public-matrix evaluations and check the resulting setup opening in the next fold. Later folds evaluate their setup contributions directly. These early folds can also afford auxiliary digits that save public bytes, because those digits remain small relative to the witness passed to the next level.

We can see this tradeoff in the outer $\mathbf{B}$ matrix. In the Hachi commitment from the warm-up, one matrix uses the inner images of all B input 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 and repeats the outer-consistency row. Commitment compression keeps that larger stack from simply reappearing as proof bytes. The slice count and transmission format must therefore be chosen together.

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, reducing expanded verifier matrix storage from 22 to 4 MiB. The prover instead stores 32 rather than 22 MiB of expanded setup matrix coefficients and must open the extra prefix. That opening enlarges the level-1 witness and raises the planner's proof-payload estimate from 63,453 to 66,250 bytes (Sections 12 and 13).

In the standalone PCS experiment of Section 13, verification falls from 32.2 to 14.1 ms. Opening time increases by 46.7% and the serialized proof by 4.5%, and commitment time rises from 8.42 to 10.7 seconds. These are the single-threaded Ryzen measurements at 2^{35} nominal committed bits, which correspond to our 2^{30} fp32 coefficients; Appendix I gives the measurement procedure. Verification improves at every tested size supporting this offloaded profile, starting at 2^{24} coefficients. The planner must compare the removed scan with the entire continuation needed to check its opening.

The added prover time has two sources, and only one of them is the setup opening. A single-threaded profile of this instance on a separate machine attributes about one third of the added opening time to level 1, whose sum-checks now also run over the precommitted setup prefix. This is the cost of offloading itself. The prefix has 2^{23} coefficients against 2^{30} in the input, so its share shrinks as the input grows. The remaining two thirds, and all of the added commitment time, come from the root parameters selected together with offloading. With offloading enabled, the planner first minimizes the largest setup scan left to the verifier and only then proof size and prover work (Section 12). The schedule it selects reaches the 2^{20}-coefficient scan with a different root: each input coefficient is split into five base-2^7 digits instead of four base-2^8 digits, and the fold challenges come from a 64-dimensional subring instead of a 128-dimensional one, with ten coefficients in $\{\pm 2\}$ in addition to the 31 in $\{\pm 1\}$. The fifth digit widens the root commitment matrix by 25%, which accounts for the commitment time. Computing the root fold response takes one sparse multiply-accumulate per digit and per nonzero challenge coefficient, hence $5 \cdot 41$ instead of $4 \cdot 31$ of them, a factor of about 1.65 on the largest single step of the opening. Both are linear-time costs of the selected parameters rather than of opening the setup. They are specific to this field and size: at 2^{35} bits the fp64 and fp128 profiles change opening time by $+12\%$ and -2% and commitment time by less than 2% (Table 16).

Recursive tail. As we reach smaller witnesses, fixed proof objects and every value appended to the next witness become a significant fraction of the total 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. The concrete parameters determine the transition; "early" and "tail" name cost regimes, not fixed level numbers.

Terminal fold. We stop once another recursive opening costs more than direct verification. Following LaBRADOR [18, §5.6], we omit commitments whose openings would immediately be revealed and avoid decomposing the last response. The preceding fold binds the terminal's inner images $\mathbf{t}_i = \mathbf{A}_{\mathsf{term}}\mathbf{s}_i$. The terminal fixes its partials $(e_i)_i$ before the challenges, then reveals $\mathbf{z} = \sum_i c_i\mathbf{s}_i$ and checks

$$\mathbf{A}_{\mathsf{term}}\mathbf{z} = \sum_i c_i\mathbf{t}_i.$$

It also checks the fold-consistency and scalar-opening equations and the response bound directly. We retain the opening conversion needed by the trace check, but remove the auxiliary commitments, quotient digits, range and relation proofs, and next commitment. Any pending setup opening must already have been checked. In the commitment and fold-consistency checks, the last block's $\mathbf{t}_i$ and e_i each appear once, multiplied by an invertible challenge. The prover therefore omits both and sends a digest of each; the verifier solves the two equations for them and compares the results with the digests. Section 8.2 specifies these direct checks; we next explain the choices that determine when they become cheaper.

2.5 Two Opening Representations

Evaluation Trace Diamond and Posen's tensor reduction [38, Theorem 3.5] turns the base-field claim into $h = \theta g(\rho)$ at a new point ρ, where g packs groups of k base-field Boolean evaluations into E-values and $\theta \neq 0$ is checked by the verifier (the reduction is the identity for $k = 1$; see Section 3.7).

We now need to recover this field evaluation from ring elements that can be folded with full-ring challenges. Put $m = d_A/k$ and choose the trace-compatible packing $\psi : E^m \to R_{q,d_A}$ and multiplicative embedding $E \hookrightarrow R_{q,d_A}$ of Hachi, generalizing LNP22 [79,75]. Index the Boolean evaluations of g by $(i, x, u) \in [B] \times [M] \times [m]$, using the scheduled packing basis, so $F_{i,x} = \psi((g_{i,x,u})_u)$. Split ρ into block, position, and packed-coordinate parts. Evaluating the position coordinates leaves one packed vector per block:

$$\begin{aligned} p_{i,u} &:= \sum_x \widetilde{\mathsf{eq}}(\rho_{\mathsf{pos}}, x) g_{i,x,u}, \\ e_i &:= \sum_x \widetilde{\mathsf{eq}}(\rho_{\mathsf{pos}}, x) F_{i,x} = \psi((p_{i,u})_u). \end{aligned} \tag{21}$$

Because the weights embed into the ring, this partial evaluation commutes with multiplication by any ring challenge. We have not yet evaluated the u coordinates: their desired contribution is $\sum_u \widetilde{\mathsf{eq}}(\rho_{\mathsf{pack}}, u) p_{i,u}$.

The trace identity computes exactly this inner product. Let H be the automorphism subgroup fixing the embedded field E, write $\mathrm{Tr}_H = \sum_{\sigma \in H} \sigma$, and let σ_{-1} send X to X^{-1}. Hachi's packing satisfies

$$\frac{1}{m} \mathrm{Tr}_H\big(\psi(a)\sigma_{-1}(\psi(b))\big) = \langle a, b \rangle. \tag{22}$$

Pack the remaining equality weights as $\check{\chi} = \psi((\widetilde{\mathsf{eq}}(\rho_{\mathsf{pack}}, u))_u)$ and set $\mathcal{T}(Z) = m^{-1}\mathrm{Tr}_H(Z\sigma_{-1}(\check{\chi}))$. Substituting $a = (p_{i,u})_u$ gives $\mathcal{T}(e_i) = \sum_u \widetilde{\mathsf{eq}}(\rho_{\mathsf{pack}}, u) p_{i,u}$, hence

$$\theta \sum_i \widetilde{\mathsf{eq}}(\rho_{\mathsf{blk}}, i) \mathcal{T}(e_i) = \theta \sum_{i,x,u} \widetilde{\mathsf{eq}}(\rho, (i, x, u)) g_{i,x,u} = h. \tag{23}$$

Hachi transmits the ring element $\sum_i \widetilde{\mathsf{eq}}(\rho_{\mathsf{blk}}, i) e_i$ and checks its trace separately. We instead prove this field equation on the committed opening digits, removing that element (4096 bytes at Hachi's parameters [79, §5.2]) and its ring-switch quotient witness. The folded evaluation relation (8) remains ring-valued.

Let b_{op} be the opening-digit base. Write $e_i = \sum_{\ell,\nu} b_{\mathsf{op}}^{\ell} \hat{e}_{i,\ell,\nu} X^{\nu}$. Linearity of $\mathcal{T}$ gives the public digit weights explicitly:

$$\begin{aligned} \tau_\nu &:= \frac{1}{m} \mathrm{Tr}_H(X^\nu \sigma_{-1}(\check{\chi})), \\ \omega_{\mathsf{Tr}}(i, \ell, \nu) &:= \theta \widetilde{\mathsf{eq}}(\rho_{\mathsf{blk}}, i) b_{\mathsf{op}}^{\ell} \tau_\nu, \\ \sum_{i,\ell,\nu} \omega_{\mathsf{Tr}}(i, \ell, \nu) \hat{e}_{i,\ell,\nu} &= h. \end{aligned} \tag{24}$$

For $k = 1$, ordinary coefficient packing gives $\tau_\nu = \widetilde{\mathsf{eq}}(\rho_{\mathsf{pack}}, \nu)$: the trace simply extracts the coefficient inner product. The same construction works for $k > 1$ through the trace-compatible packing. Place these weights at the opening-digit positions of $\mathbf{w}'$ and zero elsewhere. Then $\sum_y \widetilde{\mathbf{w}'}(y) \widetilde{\omega}_{\mathsf{Tr}}(y) = h$ over Boolean y is a degree-two sum-check claim.

Evaluating the trace weights. The verifier needs $\widetilde{\omega}_{\mathsf{Tr}}(r)$ at the final sum-check point, not the entire vector. To see the saving, consider a bit-aligned opening interval with a power-of-two block domain and a padded digit axis. Split r into interval-tag, block, digit, and coefficient coordinates. For opening depth δ, its contribution factors as

$$\begin{aligned} \widetilde{\omega}_{\mathsf{Tr}}(r) = \theta &\widetilde{\mathsf{eq}}(r_{\mathsf{tag}}, \mathsf{tag}) \widetilde{\mathsf{eq}}(\rho_{\mathsf{blk}}, r_{\mathsf{blk}}) \\ &\cdot \left(\sum_{\ell < \delta} b_{\mathsf{op}}^{\ell} \widetilde{\mathsf{eq}}(r_{\mathsf{digit}}, \ell) \right) \left(\sum_{\nu < d_A} \tau_\nu \widetilde{\mathsf{eq}}(r_{\mathsf{coef}}, \nu) \right). \end{aligned} \tag{25}$$

The block sum collapses because $\sum_i \widetilde{\mathsf{eq}}(\rho_{\mathsf{blk}}, i) \widetilde{\mathsf{eq}}(r_{\mathsf{blk}}, i) = \widetilde{\mathsf{eq}}(\rho_{\mathsf{blk}}, r_{\mathsf{blk}})$. After ring-local preprocessing of τ, evaluation therefore takes $O(\log |\mathbf{w}'| + \delta + d_A)$ field work rather than a scan of $B\delta d_A$ digits. Shifted intervals and tightly packed digit axes use the address-carry calculation in Appendices A and B.2.

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 Our evaluation-trace partials retain a full A-ring element for each block. When these partials contribute substantially to the next witness, we would like to make them shorter. We can do so by evaluating some of the coefficient coordinates before folding, provided the challenges preserve the coordinates we retain. This also lets us work directly with the original base-field polynomial, avoiding the tensor reduction and trace packing just described.

We choose powers of two d_A, d_{car}, h satisfying

$$d_A = khd_{\mathsf{car}}, \qquad k = [E : \mathbb{F}_q].$$

Our goal is a partial with $kd_{\mathsf{car}} = d_A/h$ base-field coordinates, a factor h shorter than a full A-ring element. To obtain it, we will retain d_{car} coefficients over E from each block.

Evaluating some coordinates and retaining others. Write a coefficient index uniquely as $\nu = a + khj$, with $a \in [kh]$ and $j \in [d_{\mathsf{car}}]$. We use this arrangement for each source ring element at position x of block i:

$$F_{i,x}(X) = \sum_{a\in[kh]} \sum_{j\in[d_{\mathsf{car}}]} f_{i,x,a,j} X^{a+khj} \in R := \mathbb{F}_q[X]/(X^{d_A}+1).$$

The opening point supplies equality weights for the position x and the coefficient index a. We sum over these two indices, leaving j as the exponent of a new variable Y. This gives one partial per block:

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

Each of the d_{car} coefficients lies in E, so the partial has $d_{\mathsf{op}}^{\mathsf{dir}} = kd_{\mathsf{car}}$ base-field coordinates, as desired. Summing its coefficients and the blocks with the remaining opening weights recovers the original claimed evaluation.

Folding with subring challenges. We still need these shorter partials to commute with folding. Let $\mathcal{L}$ denote the weighted sum defining e_i from block i. Multiplication by X^{kh} shifts the retained index j, including the sign change at $X^{d_A} = -1$, so

$$\mathcal{L}(X^{kh}F) = Y\mathcal{L}(F).$$

Multiplication by X instead changes the index a, over which we have already summed using the opening weights. In general, we cannot recover $\mathcal{L}(XF)$ from $\mathcal{L}(F)$. We therefore restrict challenges to polynomials in X^{kh}. Formally, we use the subring

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

The shift identity extends to every $c \in S$, giving $\mathcal{L}(\iota(c)F) = c\mathcal{L}(F)$. We can consequently check folding in the smaller ring:

$$\mathcal{L}\left(\sum_i \iota(c_i)F_i\right) = \sum_i c_i e_i.$$

We use the same challenge as c_i in this equation and as $\iota(c_i)$ in the A-ring commitment equations. The embedding preserves its nonzero coefficients, coefficient norms, and invertibility. Section 5.3 gives the complete coefficient arrangement and the challenge evaluations used after ring switching.

We obtain shorter partials and avoid the tensor-reduction sum-check, but must choose d_A divisible by kd_{car}. The smaller challenge ring must also contain enough admissible challenges for soundness. Increasing the number of nonzero challenge coefficients can enlarge the response bound and the Module-SIS collision radius, offsetting some of the saving. We must therefore account for both the partial width and the challenge parameters.

The running example. Let us see the saving at the root of our running example. Its 2^{30} base-field coefficients become 2^{19} A-ring elements at $d_A = 2048$, divided into $B = 2^{10}$ blocks of $M = 2^9$ positions. The original Boolean coordinates split as

$$30 = \underbrace{10}_{\text{block}} + \underbrace{9}_{\text{position}} + \underbrace{11}_{\text{A-ring coefficient}}.$$

Table 2: Opening and ring-relation choices in the running 2^{30}-coefficient fp32 schedule. N_j counts the input ring elements before block padding; each of the B_j blocks contains M_j 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_j	M_j	B_j	d_A	d_D	opening	d_{op}	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	✓

Here $k = 4$, $d_{\mathsf{car}} = 64$, and $h = 8$, so the last 11 coordinates split further into a five-bit index a and a six-bit index j. We evaluate the nine position bits and the five bits of a, retaining the six bits of j. The resulting partial has $4 \cdot 64 = 256$ base-field coordinates, compared with 2048 for evaluation trace. Across all 1024 blocks, this is 1 MiB rather than 8 MiB of raw partials, and it avoids a tensor reduction for the largest polynomial (Table 2).

At level 1, the recursive-witness group uses $d_A = 1024$ and $h = 4$, so direct packing still gives shorter partials. The setup-opening group uses the same ring dimension and also uses direct packing. 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.

Both representations are algebraically available when their compatibility conditions hold. The schedule family analyzed here follows the opening-mode policy of (160): nonterminal levels 0 and 1 use direct packing, and later levels and the terminal use evaluation trace. The running example follows this policy; its remaining parameters account for how partial sizes affect later commitments, range checks, and folds.

2.6 Checking Ring Relations with or without Quotients

Both opening methods leave us with equations in a polynomial quotient ring. To check them by field sum-check, we 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 correct this either by supplying a quotient polynomial or by reducing each public multiplier before forming its weights on the witness coefficients.

Write one ring equation of dimension d as

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

Here Occ indexes the terms, the A_c and Y are public, the W_c are committed witness segments, and $\circledast_d$ denotes multiplication modulo $X^d + 1$.

Checking with a quotient. Following the HMZ ring-switching technique used by Hachi [53,79], we can ask the prover to supply Q satisfying the polynomial identity

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

We check this identity at a random $\alpha \in E$. The quotient accounts for the multiples of $X^d + 1$ removed by ring multiplication. The weight on each coefficient $w_{c,j}$ of W_c then factors as $A_c(\alpha)\alpha^j$; the coefficient of X^j in Q

receives weight $-(\alpha^d+1)\alpha^j$. Both contributions enter the sum-check, with the appropriate gadget and basis factors after digit decomposition. This factorization helps both the sum-check prover and the evaluation of public weights for setup offloading. We pay for it by passing the digits of Q to 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 [56]. For each public multiplier A and coefficient position j, define the weight

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

Since $W_c(X) = \sum_j w_{c,j}X^j$, linearity turns our ring equation into

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

The sign changes from reduction modulo X^d+1 are already included in these weights, so no private quotient is needed. The main body calls this the *transpose-convolution check*: we move the public multiplication and evaluation onto the weights of the private coefficients.

We can compute all d weights for a fixed A in linear time, without separately multiplying and reducing a polynomial for each j. Appendix B.1 gives the recurrence and its connection to quotient lifting; Section 7.5 explains how the verifier evaluates the resulting public-weight polynomial.

The two methods check the same ring equation. Quotient lifting tests a polynomial residual of degree below $2d$, whereas the direct check tests the reduced residual, of degree below d. A false row therefore passes the evaluation check at α with probability at most $(2d-1)/|E|$ in the first method and $(d-1)/|E|$ in the second. Both use the same subsequent row-batching and sum-check challenges.

Choosing across the recursion. We have exchanged private quotient digits for more involved public weights. At the early folds, the factored weights of quotient lifting help us process the largest witnesses and, when offloading setup, evaluate the setup weights succinctly. Later, removing a quotient also removes its digits from every subsequent range check, commitment, and fold. That saving can outweigh the extra work of evaluating the quotient-free relation.

Our running schedule illustrates this change. Levels 0 and 1 use quotient lifting; levels 2 through 5 use the quotient-free check, and the terminal checks the remaining ring equations directly (Table 2). We choose this transition by comparing complete schedules, including the cost of the witness passed forward, rather than minimizing the work of a single fold.

2.7 Commitment Compression

We have reduced the private material carried into later folds. The public commitment images can still occupy a substantial part of the proof, however, even after the witness has become small. Following our warm-up, an uncompressed fold sends the opening image $\mathbf{v} = \mathbf{D}\hat{\mathbf{e}}$ and the sliced outer image $\mathbf{u}$. Their respective coefficient counts are $s_D = n_D d_D$ and $s_B = S_B n_B d_B$, so together they occupy

$$\left\lceil \frac{\kappa}{8} \right\rceil (s_B + s_D) \tag{26}$$

bytes over a κ-bit field. These sizes depend on the commitment ranks and ring dimensions and need not decrease at the same rate as the witness.

We reduce these images by committing to their binary decompositions. This keeps an algebraic connection to the original commitment equations: the recursive relation checks both the recomposition of the digits and the new commitment to them. We apply the construction independently to the complete sliced $\mathbf{B}$-image and the unsliced $\mathbf{D}$-image.

Two stages of re-commitment. We use two maps, both applied to binary decompositions. Each map is sized for the complete digit vector it receives. We choose the two-element alphabet $\{-1, 0\}$ so that its range constraint can be combined with the existing range-check sum-check. Let $\mathbf{y}_0 \in \mathbb{F}_q^{s_0}$ denote the coefficient vector of either raw image. We write every coefficient in negative binary,

$$y = \sum_{\ell=0}^{\kappa-1} 2^\ell \xi_\ell \pmod q, \qquad \xi_\ell \in \{-1, 0\}, \tag{27}$$

and repack the resulting κs_0 digits as a vector $\hat{\mathbf{y}}_1$ over a smaller ring R_{d_1}. The corresponding gadget matrix $\mathbf{G}_1$ recomposes the original coefficients, while a public setup matrix $\mathbf{K}_1$ of module rank n_1 commits to the same digits:

$$\mathbf{y}_0 = \mathbf{G}_1 \hat{\mathbf{y}}_1, \qquad \mathbf{y}_1 = \mathbf{K}_1 \hat{\mathbf{y}}_1 \in R_{d_1}^{n_1}. \tag{28}$$

We apply this step again to $\mathbf{y}_1$ and transmit only the final payload $\mathbf{p}$. Both digit vectors join the recursive witness, and the relation sum-check authenticates both links; the verifier never recovers the hidden images. We call the outer chain $\mathbf{F}_1, \mathbf{F}_2$ and the opening chain $\mathbf{H}_1, \mathbf{H}_2$ (Equation (71)).

The digit alphabet also gives the binding argument its short-opening premise: two legal negative-binary openings differ coefficientwise by at most one. We therefore price each $\mathbf{F}$- and $\mathbf{H}$-map as a standalone Module-SIS instance with $\|\Delta\hat{\mathbf{y}}_h\|_\infty \le 1$. To justify this bound, every compression digit is constrained by $w(w+1) = 0$. This binary constraint 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. The intermediate image bounds the input to the final map: one small rank-one map cannot securely absorb arbitrary source lengths. For 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 analogous 64- and 128-bit profiles also end at 128 bytes; Table 5 gives the widths and certified cutoffs. Larger inputs require sizing the maps again and need not retain that output size.

The saving adds work to the next fold. For a 32-bit raw image with s_0 coefficients, its chain contributes

$$32s_0 + 2048 \tag{29}$$

private binary coordinates: digits of the raw image and of the 64-coefficient intermediate image. It also adds two map relations and, when quotient lifting is active, their quotient material. We evaluate these small maps directly even when the main setup scan is offloaded.

Near the root, these additions are small beside the remaining witness. Near the tail, processing them through further folds can cost more than sending the raw image. Our running schedule therefore compresses levels 0–4 and sends raw images at level 5; levels 2–4 already use quotient-free checks, so the two transitions are chosen separately. Akita permits one transition to raw images, after which compression does not resume; root and setup-prefix commitments remain compressed. With G groups, there are G outer chains and one shared opening chain, all in the same mode. The next optimization instead reduces the bound certified for the response itself.

2.8 Bounding Response Coefficients and the Total Squared Norm

We next return to the folded response itself. Its digit-range proof already bounds every coefficient, but this need not give the squared-norm bound we want for the binding analysis. Write $N_A = m_A d_A$ for the number of response coefficients used by $\mathbf{A}$. The certified coefficient bound gives

$$|z_u| \le Z_\infty \quad \text{for every } u \in [N_A], \qquad \|\mathbf{z}\|_2^2 \le N_A Z_\infty^2. \tag{30}$$

It permits all N_A coefficients to approach Z_∞ in magnitude at once. An honest-response model may predict that this is unlikely, but the verifier must also account for adversarial responses, which need not follow that model.

Certifying the sum of squares. We can enforce a tighter bound by proving the additional statement

$$\mathcal{E}_{\mathsf{resp}} := \sum_{u=0}^{N_A-1} z_u^2 \le S_{\max}. \tag{31}$$

The digit-range proof still controls each coefficient; the new check limits how large they can be collectively. When $S_{\max} < N_A Z_\infty^2$, it can reduce the Module-SIS parameters needed for $\mathbf{A}$. This gain remains even when both bounds are chosen from the same model. For independent Gaussian coefficients, for example, the coefficient-wise bound can lose a logarithmic factor in squared radius. Appendix G.3 works out this illustration; the protocol enforces the chosen bounds regardless of the distribution that motivated them.

Checking a squared norm that does not fit in the field. We have already committed to the digits of $\mathbf{z}$, so we prove its squared norm from those digits and merge it with the final range-check sum-check. If the largest possible square sum is below q, one field identity certifies the integer equality

$$\sum_u z_u^2 = S. \tag{32}$$

In the 32-bit field of our running example, however, the complete sum can exceed q at the relevant tail dimensions, so equality in $\mathbb{F}_q$ would reveal only its residue. Write $z_u = \sum_h b^h z_{u,h}$ and partition the coordinates into public segments I_t. We prove the smaller inner products

$$p_{t,h,k} := \sum_{u \in I_t} z_{u,h} z_{u,k}, \tag{33}$$

each of which is small enough to lift uniquely from $\mathbb{F}_q$ to $\mathbb{Z}$. The verifier then reconstructs the full square sum over the integers as

$$S = \sum_t \left(\sum_h b^{2h} \bar{p}_{t,h,h} + 2 \sum_{h<k} b^{h+k} \bar{p}_{t,h,k} \right). \tag{34}$$

Here $\bar{p}_{t,h,k}$ is the unique centered integer lift. The verifier then checks $S \le S_{\max}$ over the integers. This *digit-expanded norm reconstruction* preserves the Euclidean guarantee even when the field cannot contain the complete norm (Section 6.2).

Separately controlling challenge multiplication. A smaller response norm helps the binding analysis, but we must also bound how much multiplication by a challenge can enlarge that response. Let κ_1 bound the coefficient ℓ_1 norm of a folding challenge. Young's inequality gives

$$\|c\mathbf{z}\|_2 \le \kappa_1 \|\mathbf{z}\|_2. \tag{35}$$

Before invoking a particular Module-SIS estimator, we can therefore isolate the two improvements as the following reductions in squared collision radius:

$$64\kappa_1^2 N_A Z_\infty^2 \quad \longrightarrow \quad 64\kappa_1^2 S_{\max} \quad \longrightarrow \quad 64\Gamma^2 S_{\max}. \tag{36}$$

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. Thus Γ bounds how much challenge multiplication stretches a vector. These gains are independent: an ℓ_2 certificate helps without challenge rejection, whereas operator-norm rejection alone retains the box factor $N_A Z_\infty^2$. We can use either improvement or both. The coefficient path instead prices 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 D).

Choosing bounds for honest responses. The verifier enforces the selected bounds on every accepted response; we must separately ensure that an honest prover can meet them. At the root, we use deterministic envelopes or probabilistic bounds over fresh challenge signs for every source satisfying the public conditions. Filtering changes the challenge distribution, so its effect on these bounds must be included (Appendix G.1).

For later folds, our Gaussian-inspired model proposes encoding thresholds and squared-norm bounds. An encoding threshold determines the digit depth; the range check can certify a wider coefficient envelope. We do not assume this source model for arbitrary root inputs, and its predictions for later responses are not proved after conditioning on the accepted transcript. Those parameters retain the history-conditional completeness premise of Appendix G.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 G.2 and G.4.

We compare the smaller $\mathbf{A}$ image obtained from tighter bounds with the norm-proof and retry costs across the remaining recursion. This is why parameter selection must consider complete schedules rather than choose each response bound in isolation.

2.9 Batching and Distributed Proving

An application such as Jolt opens several commitments, at different points and dimensions. We build on Greyhound's batching pattern [81, §3.2]: retain separate responses while sharing 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.

We use one opening representation for the batch. For G groups, let $n_{\mathsf{clm},j}$ count the claims in group j and B_j its input blocks per claim. We retain its source geometry and use its own A-ring folding challenges $c^A_{j,c,i}$ to form the response

$$\mathbf{z}_j := \sum_{c\in[n_{\mathsf{clm},j}]} \sum_{i\in[B_j]} c^A_{j,c,i}\mathbf{s}_{j,c,i}. \tag{37}$$

We check it against its own $\mathbf{A}_j$, but bind all partial digits with one unsliced map,

$$\mathbf{v} = \mathbf{D}\,\mathsf{concat}(\hat{\mathbf{e}}_0,\dots,\hat{\mathbf{e}}_{G-1}). \tag{38}$$

When compression is active, one shared $\mathbf{H}$ chain compresses that image. After binding the partials, fresh scalar weights combine the opening claims. One relation argument checks the combined equations, and one range argument certifies their witness digits. Trace openings first use the shared tensor reduction; packing openings combine their padded claimed values directly. Admissible padding must have a nonzero public factor so that it does not erase a claim. Sections 5.1 and 5.5 give the precise input conditions and the one successor witness.

To support commitments formed before their batch is known, we fix the common accepted 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 retains its block length, rings, ranks, slices, and compression path, and the later fold must choose compatible response and opening parameters (Section 4.6). Different points remain distinct throughout. This same mechanism checks a setup-prefix opening beside the recursive witness, completing the offloading step described earlier.

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

$$\mathbf{z}^{(j)} = \sum_{i\in\mathcal{I}_j} c_i\mathbf{s}_i, \qquad \mathbf{z} = \sum_j \mathbf{z}^{(j)}. \tag{39}$$

Each $\mathbf{z}^{(j)}$ still has the full response length. Forming $\mathbf{z}$ therefore requires exchanging large vectors, and local digit decompositions do not avoid the exchange: carries prevent their sum from being the canonical digits of $\mathbf{z}$.

We instead retain and decompose each local response. The next witness contains every $\hat{\mathbf{z}}^{(j)}$, while the opening and inner-image digits partition their original vectors. With equal response digit depths, and excluding shared compression and quotient material, its length is

$$|\mathbf{w}_{\mathsf{chunk}}| = N_{\mathsf{chunk}}|\hat{\mathbf{z}}| + |\hat{\mathbf{e}}| + |\hat{\mathbf{t}}|. \tag{40}$$

Thus avoiding the exchange adds $(N_{\mathsf{chunk}} - 1)|\hat{\mathbf{z}}|$ coordinates. Linearity lets the relation check the aggregate effect of these responses, and their public weights can be combined before one setup scan. The schedule returns to a single response when carrying the extra digits becomes more expensive than consolidating them (Section 11).

The partition is fixed before the challenges; it need not match the physical machine assignment. One nonce must make all responses admissible, so completeness accounts for joint failure, without assuming independence. Binding uses the sum of their certified envelopes for the effective response $\mathbf{z}_\Sigma$. The analyzed schedule uses coefficient bounds until one response remains. 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

We have now seen why the parameters of one fold cannot be chosen independently of its successors. We make these choices before proving, producing a complete schedule that the prover and verifier can reuse. Our planner compares complete continuations, prioritizing verifier efficiency, then proof size and prover work.

For each admitted root shape, we construct a complete *schedule*

$$\mathcal{O}_0 \xrightarrow{\mathsf{Fold}_0} \mathcal{O}_1 \xrightarrow{\mathsf{Fold}_1} \cdots \xrightarrow{\mathsf{Fold}_{L-1}} \mathcal{O}_L \xrightarrow{\mathsf{Terminal}} \mathsf{accept}. \tag{41}$$

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. Existing commitment parameters remain 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 collects the background used throughout Akita: the rings, fields, and challenge distributions underlying the commitment scheme; the sum-check, ring-switching, and tensor reductions that connect ring identities to polynomial evaluations; and the commitment and special-soundness notions used in the security proof. We use $[n] := \{0, \ldots, n-1\}$ for zero-based index sets.

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 $R_q := \mathbb{Z}_q[X]/(X^d + 1)$. (We write $\log d$ for the base-2 logarithm; throughout, α is reserved for the ring-reduction challenge used by both ring-checking routes of Section 7.) An element $w \in R_q$ has coefficient vector $\mathsf{cf}(w) = (w_0, \ldots, w_{d-1}) \in \mathbb{Z}_q^d$.

Partial splitting. 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}:

$$X^d + 1 \equiv \prod_{j=1}^{s_{\mathsf{split}}} (X^{d/s_{\mathsf{split}}} - r_j) \pmod{q} \tag{42}$$

for distinct $r_j \in \mathbb{Z}_q^\times$ [76, Corollary 1.2], giving $R_q \cong \prod_{j=1}^{s_{\mathsf{split}}} \mathbb{F}_{q^{d/s_{\mathsf{split}}}}$ by CRT. The case $s_{\mathsf{split}} = 2$ (i.e. $q \equiv 5 \pmod 8$) is the minimal splitting used in most prior work [18,81,79]; larger s_{split} permits faster NTT-based arithmetic at the cost of a tighter invertibility threshold.

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 ($s_{\mathsf{split}} = 2$), $H = \langle \sigma_{-1}, \sigma_{4k+1} \rangle$ 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 $\mathbf{a}, \mathbf{b} \in (\mathbb{F}_{q^k})^{d/k}$ [79, Theorem 2].

Ring-subfield coordinates. In the minimal-splitting case $s_{\mathsf{split}} = 2$ ($q \equiv 5 \pmod 8$), the trace embedding above has the following concrete ring-subfield basis for the power-of-two extension degrees used below. Let k be such an extension degree and set $m := d/(2k)$. Inside R_q, define

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

The fixed subfield has basis $\{1, e_1, \ldots, e_{k-1}\}$ over $\mathbb{F}_q$. Using $X^d = -1$, this may also be written as

$$e_j = X^{jm} - X^{d-jm}.$$

This is the coordinate system used by the trace embedding above. The Akita Book gives the corresponding optimized arithmetic for the degree-2 and degree-4 cases [64].

Base-field coefficients and extension-field points. We distinguish two uses of a field extension $E := \mathbb{F}_{q^k}$: it may be the coefficient field of the committed polynomial, or only the challenge and evaluation field used by sum-check. The latter case is common in Akita: we often commit to a polynomial through its $\mathbb{F}_q$-valued Boolean evaluations, but evaluate it at points in E to obtain negligible sum-check soundness.

The ordinary coefficient embedding is the $k = 1$ degeneration of the same trace construction.[11] When $E = \mathbb{F}_q$, the fixed subring is just the coefficient field and ψ is the usual coefficient-packing map

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

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 $\sum_i a_i b_i$, and the trace identity above is exactly a scaled constant-term extraction.

Coefficient norms and challenges. For $w \in R_q$ we write $\|w\|_\infty := \max_j |w_j \bmod^{\pm} q|$, using centered representatives in $(-q/2, q/2]$. We also write $\|w\|_1 := \sum_j |w_j \bmod^{\pm} q|$ and $\|w\|_2 := (\sum_j |w_j \bmod^{\pm} q|^2)^{1/2}$. For a vector $\mathbf{w} \in R_q^m$, each norm is taken over the concatenation of all md centered coefficient coordinates.

For $a \in R_q$, let $\mathsf{M}_a \in \mathbb{Z}^{d \times d}$ be the integer negacyclic-convolution matrix of the centered coefficient lift of a. For $\mathbf{v} = (v_1, \ldots, v_m) \in R_q^m$, define

$$\mathsf{M}_{\mathbf{v}} := \begin{pmatrix} \mathsf{M}_{v_1} \\ \vdots \\ \mathsf{M}_{v_m} \end{pmatrix}, \qquad \|\mathbf{v}\|_{\mathsf{mul},2} := \|\mathsf{M}_{\mathbf{v}}\|_{2\to 2}.$$

Thus $\|a\|_{\mathsf{mul},2}$ is the spectral norm of multiplication by a ring element, while $\|\mathbf{v}\|_{\mathsf{mul},2}$ is the spectral norm of the map $a \mapsto a\mathbf{v}$. Multiplication by a acts block-diagonally on a vector in R_q^m, so its operator norm there remains $\|a\|_{\mathsf{mul},2}$ and does not acquire a factor depending on m. Writing $\zeta_k := e^{(2k+1)\pi i/d}$, unitary diagonalization of negacyclic convolution gives the exact formulas

$$\|a\|_{\mathsf{mul},2} = \max_{k \in [0,d)} |a(\zeta_k)|, \qquad \|\mathbf{v}\|_{\mathsf{mul},2} = \max_{k \in [0,d)} \left(\sum_{j=1}^{m} |v_j(\zeta_k)|^2 \right)^{1/2}.$$

These identities let the protocol bound the challenge multiplication norm by deterministic rejection sampling, provided the accepted family still meets the special-soundness size requirement below. They also let the verifier compute the response multiplication norm directly when the response is revealed.

The following inequalities give the three norm pairings used in the binding proof.

[11] For $k > 1$, a coefficient-wise encoding of an element of $\mathbb{F}_{q^k}$ into several coefficient slots of R_q is only $\mathbb{F}_q$-linear: the chosen coefficient span is not generally closed under ring multiplication, so it does not preserve multiplication by extension-field evaluation points. The trace construction therefore uses a multiplicative subfield $R_q^H \simeq \mathbb{F}_{q^k}$ and recovers the packed inner product via the trace map.

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

$$\|a\mathbf{v}\|_\infty \le \|a\|_1 \|\mathbf{v}\|_\infty, \tag{43}$$
$$\|a\mathbf{v}\|_2 \le \min\{\|a\|_1\|\mathbf{v}\|_2, \|a\|_{\mathsf{mul},2}\|\mathbf{v}\|_2, \|a\|_2\|\mathbf{v}\|_{\mathsf{mul},2}\}. \tag{44}$$

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 the block-diagonal operator M_a to the coefficient vector of $\mathbf{v}$. Commutativity lets us instead apply $\mathsf{M}_{\mathbf{v}}$ to the coefficient vector of a, which gives the third Euclidean bound. Centered reduction modulo q cannot increase the absolute value of any coefficient, so all these bounds remain valid in R_q.

For the $k = 1$ coefficient embedding, ψ preserves coefficient norms. For $k > 1$ in the minimal-splitting construction, write $m = d/(2k)$ and use the subfield basis $1, e_1, \dots, e_{k-1}$ above [79, Lemma 4]. The packing map of [79, Theorem 2] combines two banks of m extension elements:

$$\psi(\mathbf{a}, \mathbf{b}) = \sum_{t=0}^{m-1} X^t (a_t + X^{d/2} b_t), \qquad a_t = a_{t,0} + \sum_{j=1}^{k-1} a_{t,j} e_j, \tag{45}$$

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

$$\begin{aligned} c_t &= a_{t,0}, & c_{t+d/2} &= b_{t,0}, \\ c_{t+jm} &= a_{t,j} + b_{t,k-j}, & c_{t+d-jm} &= -a_{t,j} + b_{t,k-j} \quad (1 \le j < k). \end{aligned} \tag{46}$$

The two banks therefore overlap: the coefficient ℓ_∞ norm can increase by a factor of 2. The coefficient ℓ_1 norm also grows by 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 can only decrease these norms. These are logical-to-ring conversion factors, applied only when a bound is stated before ψ. A norm stated directly 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 in the ring-coefficient space used by Module-SIS and receives no further embedding factor.

When $X^d + 1$ splits into s_{split} factors as in (42), every nonzero $c \in R_q$ with $\|c\|_\infty < q^{1/s_{\mathsf{split}}}/\sqrt{s_{\mathsf{split}}}$ is invertible [76, Corollary 1.2]; for $s_{\mathsf{split}} = 2$ this gives the familiar bound $\|c\|_\infty < q^{1/2}/\sqrt{2}$. Concrete families used by Akita are specified in Section 3.2.

3.2 Sparse and Filtered Folding Challenges

A folding step samples challenges from a finite set $\mathcal{C} \subset R_q$. A challenge is supported on a small random subset of coefficient positions, with independent random signs and fixed magnitudes on that support. The family is fixed by the ring dimension d and the counts $(n_a)_{a\in\mathcal{A}}$, where $\mathcal{A} \subset \mathbb{Z}_{>0}$ is a finite set of allowed magnitudes and each $n_a \ge 0$ counts how many support coordinates have absolute value a. Define the nonempty active set $\mathcal{A}_+ := \{a \in \mathcal{A} : n_a > 0\}$ and require $1 \le s := \sum_{a\in\mathcal{A}} n_a \le d$. Write

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

A challenge is drawn by the following three steps.

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 $\|c\|_1 = \omega$. The family has cardinality

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

where $\binom{s}{n_a : a\in\mathcal{A}}$ is the multinomial coefficient $s!/\prod_a n_a!$ when $|\mathcal{A}| > 1$ and equals 1 when $|\mathcal{A}| = 1$. Write $\mathcal{C}^{\mathsf{acc}}$ for the family that the verifier samples after any fixed, witness-independent filter; without a filter, $\mathcal{C}^{\mathsf{acc}} = \mathcal{C}$.

A fold with W challenge coordinates has coordinate-wise special-soundness error $W/|\mathcal{C}^{\mathsf{acc}}|$ (Section 3.8). For extraction, any challenge difference $\bar{c} = c - c'$ obeys $\|\bar{c}\|_1 \le 2\omega$ because each coordinate of $\bar{c}$ is a difference of two values in $[-c_{\max}, c_{\max}]$ with at most one nonzero per side. When $X^d + 1$ splits as in (42), every nonzero such difference with $\|\bar{c}\|_\infty \le 2c_{\max}$ is 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 sparse family $\mathcal{C}_0$ as above and a public threshold Γ, the accepted family is

$$\mathcal{C}_\Gamma^{\mathsf{acc}} := \{c \in \mathcal{C}_0 : \mathsf{OpNormAccept}_\Gamma(c) = 1\}, \tag{47}$$

where $\mathsf{OpNormAccept}_\Gamma$ is a deterministic, witness-independent predicate that accepts only challenges with $\|c\|_{\mathsf{mul},2} \le \Gamma$. The protocol samples from $\mathcal{C}_0$ and rejects elements outside $\mathcal{C}_\Gamma^{\mathsf{acc}}$, so its output is uniform over $\mathcal{C}_\Gamma^{\mathsf{acc}}$. Appendix D defines the predicate, proves this norm bound (Theorem D.1), and certifies a lower bound on $|\mathcal{C}_\Gamma^{\mathsf{acc}}|$, against which the soundness error is charged.

Sampling. The three-step rule above 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 s independent sign bits. In the noninteractive protocol these draws come from a domain-separated XOF stream. The number of bits read is therefore variable rather than exactly $\lceil \log_2 n \rceil$ for an interval of size n; the Akita Book specifies the byte-level procedure [64].

Rademacher sums and Hoeffding bounds. In several of our checks, 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. Rademacher sums and Hoeffding's inequality quantify the cancellation.

Definition 3.2 (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*

$$V_X := \sum_{j\in J} w_j^2.$$

In applications the weights are often deterministic only after conditioning on some public or previously sampled data. We use the following convention: after such conditioning, if the remaining random variables are independent Rademachers and the coefficients of those signs are fixed, the conditioned quantity is a Rademacher sum. Tail bounds proved for every conditioning event then also hold unconditionally by averaging over the conditioning.

Lemma 3.3 (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 $t \ge 0$,

$$\Pr[|X| > t] \le 2\exp\left(-\frac{t^2}{2V_X}\right),$$

with the convention that the right-hand side is 0 when $V_X = 0$ (for every $t \ge 0$).

Lemma 3.3 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 this bound to size digit intervals from squared weights.

3.3 Gadget Decomposition and Module-SIS

Fix a power-of-two decomposition base $b = 2^\ell > 1$ 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 chosen right inverse of $\mathbf{G}_{b,n}$ that decomposes each coefficient into balanced base-b digits. A balanced digit lies in the half-open interval

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

Thus a k-digit balanced expansion has the form

$$x = \sum_{i=0}^{k-1} d_i b^i, \qquad d_i \in \mathcal{D}_b.$$

Compared with unsigned digits in $[0, b)$, this reduces the digit ℓ_∞ bound from $b-1$ to $b/2$ and the worst-case squared ℓ_2 digit norm from $k(b-1)^2$ to $k(b/2)^2$.

Because $\mathcal{D}_b$ is asymmetric, the exact representable range with k digits is also asymmetric. Its largest positive value is

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

and its largest negative magnitude is

$$M_k := \frac{b}{2} \cdot \frac{b^k - 1}{b - 1}.$$

The interval $[-M_k, T_k]$ contains exactly b^k integers. For full-field decompositions we set $k = \lceil \log_b q \rceil$ and lift a canonical residue $x \in [0, q)$ to an integer $\tilde{x}$ using the threshold

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

This guarantees $\tilde{x} \in [-M_k, T_k]$, so no extra digit is needed.

Different protocol components may use different bases. If b is the commitment base and b_1 is the opening base, we write $\delta_{\mathsf{com}} := \lceil \log_b q \rceil$ and $\delta_{\mathsf{open}} := \lceil \log_{b_1} q \rceil$ for the commitment- and opening-level depths. When a single base is active in a local argument, we may abbreviate $\mathbf{G}_{b,n}$ and $\mathbf{G}_{b,n}^{-1}$ as $\mathbf{G}_n$ and $\mathbf{G}_n^{-1}$. We keep the base explicit whenever one display mixes commitment, opening, quotient, or fold-witness decompositions, and in all root-relation rows where the dimensions alone do not identify the base.

Definition 3.4 (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 $\|\mathbf{z}\|_{p,\mathrm{coef}} \le \beta$. Here $\|\cdot\|_{p,\mathrm{coef}}$ is the ℓ_p norm of the concatenated centered coefficient vector. We omit the superscript and write $\mathsf{MSIS}_{q,d}$ for the coefficient-ℓ_∞ problem.*

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. The three valid challenge-response pairings are exactly those of Lemma 3.1. Hardness targets and the offline estimators are documented in Appendix C.

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 $i \mapsto f_i$.

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 $T_j = s_j(0) + s_j(1)$,
$$s_{j,1} = T_j - 2s_{j,0} - \sum_{i\ge 2} s_{j,i}.$$
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 [73]: each round message omits its linear coefficient, which the verifier recovers from the running claim $T_j = s_j(0) + s_j(1)$.

Theorem 3.5 (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 $\mu\ell/|\mathbb{F}|$ [73].*
2. $(\ell+1)$-special soundness. *Fix the prefix and the prover's round-j message. If $\ell+1$ distinct challenges $r_j^{(1)}, \dots, r_j^{(\ell+1)}$ have continuations certifying the corresponding honest residual sums, then interpolation identifies the round polynomial with the honest degree-at-most-ℓ polynomial s_j [79, 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 $(1, \dots, 1)$-coordinate-wise $(\ell+1, \dots, \ell+1)$-special sound (Definition 3.12), with knowledge error $\mu\ell/|\mathbb{F}|$.*

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 local variables in a subset of the common round positions. We extend g_i to an M-variate polynomial by ignoring every coordinate outside this subset:

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

Every ignored coordinate doubles the Boolean sum. Thus

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

After sampling $\gamma_1, \dots, \gamma_t \leftarrow \mathbb{F}$, the verifier can therefore batch the claims as

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

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

This subset view also describes the round messages. Let $C_j^{(i)}$ be the running claim for instance i at the start of round j, with $C_1^{(i)} := 2^{M-\mu_i} T^{(i)}$. If j lies in the image of π_i, the instance is active and contributes its ordinary degree-ℓ round polynomial. Otherwise, $\hat{g}_i$ is independent of the current variable, so its round polynomial is the constant

$$s_j^{(i)}(X) = C_j^{(i)}/2.$$

Indeed, $s_j^{(i)}(0) + s_j^{(i)}(1) = C_j^{(i)}$, and after challenge r_j the next claim is $C_{j+1}^{(i)} = C_j^{(i)}/2$. The prover sends the combined round polynomial $s_j = \sum_i \gamma_i s_j^{(i)}$. At the end, the verifier checks the corresponding linear combination of

$$g_i\big(r_{\pi_i(1)}, \ldots, r_{\pi_i(\mu_i)}\big).$$

We use the conservative bound that this random linear-combination check 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 $(r_{M-\mu_i+1}, \ldots, r_M)$ of the shared challenge vector.

3.5 Equality-Factored Sum-Check

We use the equality-factored sum-check of [49,36] 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. In round j, let q_j be the degree-$\leq \ell - 1$ univariate obtained after binding the earlier coordinates and summing p against the equality factors of the later coordinates. We normalize the running claim by removing each equality factor as its coordinate is processed. Thus

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

The coefficient of $q_{j,0}$ in this identity is one. We therefore omit the constant coefficient, rather than the linear coefficient, and send $(q_{j,1}, \ldots, q_{j,\ell-1})$. The verifier recovers

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

using only multiplication and subtraction. This rule is inversion-free for every $\tau_j \in \mathbb{F}$, including $\tau_j = 0$, and sends $\ell - 1$ coefficients, one fewer than the standard degree-ℓ linear-omission format.

The factorization also removes the full vector of equality weights from the prover's inner loop [49,36]. The prover caches equality weights for the two halves of τ, using about $2^{\mu/2}$ values per half. It reads each remaining equality weight as a product of two cached values.

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 6 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 [53] and used in Hachi [79], 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. We reserve "ring switching" for this polynomial-ring lift, distinct from the tensor reduction of Section 3.7.

The lift. Since q is prime, $\mathbb{Z}_q[X]$ is Euclidean and division by the monic $X^d + 1$ 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 3.6 (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 $\alpha \in \mathbb{F}_{q^k}$, the evaluation map $\varphi_\alpha : \mathbb{Z}_q[X] \to \mathbb{F}_{q^k}$, $X \mapsto \alpha$, is a ring homomorphism, so applying it coordinatewise to Lemma 3.6 gives the scalar identity over $\mathbb{F}_{q^k}$

$$\mathsf{M}(\alpha)\,\mathbf{w}(\alpha) = \mathbf{h}(\alpha) + (\alpha^d+1)\,\mathbf{r}(\alpha), \tag{48}$$

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

$$\Pr_{\alpha \leftarrow \mathbb{F}_{q^k}}\big[\mathbf{p}(\alpha) = \mathbf{0}\big] \le \frac{2d-1}{q^k}.$$

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

$$\widetilde{\alpha}(y) = \prod_{j=0}^{\log d - 1} \big(1 + y_j(\alpha^{2^j} - 1)\big) \in \mathbb{F}_{q^k}, \tag{49}$$

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

$$\mathbf{w}_x(\alpha) = \sum_{\ell<d} \mathbf{w}(x, y(\ell))\,\alpha^\ell = \sum_{y\in\{0,1\}^{\log d}} \widetilde{\mathbf{w}}(x,y)\,\widetilde{\alpha}(y). \tag{50}$$

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 [38, 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 cyclotomic 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; the transcript and soundness error of the reduction belong to that receiving fold.

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

$$g(X_{\mathsf{tail}}) := \sum_{y\in\{0,1\}^\kappa} f(y, X_{\mathsf{tail}})\,\beta_y \in \mathbb{F}_{q^{2^\kappa}}^{\le 1}[X_{\mathsf{tail}}].$$

The packed polynomial g has $\ell - \kappa$ Boolean variables and coefficients in $\mathbb{F}_{q^{2^\kappa}}$.

Input and output instances. The *input* is an evaluation claim

$$f(r_{\mathsf{head}}, r_{\mathsf{tail}}) = v, \qquad r_{\mathsf{head}} \in \mathbb{F}_{q^{2^\kappa}}^{\kappa},\ r_{\mathsf{tail}} \in \mathbb{F}_{q^{2^\kappa}}^{\ell-\kappa},\ v \in \mathbb{F}_{q^{2^\kappa}},$$

on the base-field polynomial f. The *output* is a single evaluation claim on the packed polynomial g at a point $\rho \in \mathbb{F}_{q^{2^\kappa}}^{\ell-\kappa}$ produced by the reduction.

Column and row partials. The prover sends the *column partials*

$$S_y := f(y, r_{\mathsf{tail}}) \in \mathbb{F}_{q^{2^\kappa}}, \qquad y \in \{0,1\}^\kappa,$$

and the verifier checks the input claim by multilinear recombination over the head, $v = \sum_y \widetilde{\mathsf{eq}}(y, r_{\mathsf{head}})\, S_y$. The tensor step is what prevents the insecure shortcut of checking only $\sum_y \beta_y S_y$: the basis combination is binding for $\mathbb{F}_q$-coordinates, not for arbitrary $\mathbb{F}_{q^{2^\kappa}}$-coefficients. To bind the S_y to g over $\mathbb{F}_q$, the verifier works in the tensor algebra $\mathbb{F}_{q^{2^\kappa}} \otimes_{\mathbb{F}_q} \mathbb{F}_{q^{2^\kappa}}$, whose elements are $2^\kappa \times 2^\kappa$ matrices over $\mathbb{F}_q$ read either column-wise (a list $(\cdot)_y$ of extension elements) or row-wise (a list $(\cdot)_u$). Decomposing each column in $\mathbb{F}_q$-coordinates, $S_y = \sum_u S_{u,y}\, \beta_u$ with $S_{u,y} \in \mathbb{F}_q$, the verifier forms the *row partials*

$$row_u := \sum_{y \in \{0,1\}^\kappa} S_{u,y}\, \beta_y, \qquad u \in \{0,1\}^\kappa.$$

Let $A_u(w) \in \mathbb{F}_q$ be the coordinates of the tail equality factor, $\widetilde{\mathsf{eq}}(r_{\mathsf{tail}}, w) = \sum_u A_u(w)\, \beta_u$ for $w \in \{0,1\}^{\ell-\kappa}$, and let A_u be the multilinear extension of $w \mapsto A_u(w)$. Since $S_{u,y} = \sum_w A_u(w)\, f(y,w)$ by multilinear extension of f over the tail, the honest row partials satisfy

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

Tensor-reduction sum-check. The verifier batches the 2^κ claims $(\star)$ with a challenge $\eta \in \mathbb{F}_{q^{2^\kappa}}^{\kappa}$, $\eta_u := \widetilde{\mathsf{eq}}(u, \eta)$:

$$c_\eta := \sum_u \eta_u\, row_u, \qquad A_\eta(w) := \sum_u \eta_u\, A_u(w),$$

so that $c_\eta = \sum_{w \in \{0,1\}^{\ell-\kappa}} A_\eta(w)\, g(w)$. The two parties run sum-check (Figure 2) on the degree-2 product $A_\eta(w)\, g(w)$ over its $\ell - \kappa$ variables with initial claim c_η, yielding a point $\rho \in \mathbb{F}_{q^{2^\kappa}}^{\ell-\kappa}$ and final value $final = A_\eta(\rho)\, g(\rho)$. The factor $A_\eta(\rho)$ is transparent: the values $A_u(\rho)$ are the rows of $e := \widetilde{\mathsf{eq}}(\phi_0(r_{\mathsf{tail}}), \phi_1(\rho))$ in the tensor algebra, where $\phi_0(\alpha) := \alpha \otimes 1$ and $\phi_1(\alpha) := 1 \otimes \alpha$ are the two canonical embeddings [38]. Writing $e = \sum_u \beta_u \otimes e_u$, the verifier obtains $A_\eta(\rho) = \sum_u \eta_u\, e_u$ in $O(\ell - \kappa)$ tensor-algebra operations. The verifier rejects if $A_\eta(\rho) = 0$. Otherwise it checks $final = A_\eta(\rho)\, g(\rho)$ and outputs the single evaluation claim $g(\rho)$. The nonzero check is necessary: if $A_\eta(\rho) = 0$, the equality holds for every value of $g(\rho)$ and authenticates none. An honest prover is therefore rejected with the probability that $A_\eta(\rho) = 0$. The factor $A_\eta(\rho)$ is jointly polynomial in η and ρ, with degree at most κ in the former and $\ell - \kappa$ in the latter. Hence an honest prover is rejected with probability at most $\ell/|\mathbb{F}_{q^{2^\kappa}}|$, and a false input claim passes with probability at most $(\kappa + 2(\ell - \kappa))/|\mathbb{F}_{q^{2^\kappa}}|$. Theorem 5.1 proves these bounds for the batched form of the reduction that Akita uses.

3.8 Polynomial Commitments and Coordinate-Wise Special Soundness

We first fix the polynomial-commitment syntax and security notions, and then recall the coordinate-wise special-soundness framework. The extensions of this framework that our proof requires are developed in Section 10.2.

Definition 3.7 (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.*

Stating knowledge soundness through $\mathsf{Rel}_{\mathsf{com}}$ avoids identifying a valid decommitment with a fresh execution of a potentially randomized commitment algorithm. A concrete scheme may also use an algebraic decommitment relation that contains the canonical state produced by its commitment algorithm.

For later use, define the evaluation relation

$$\mathsf{Rel}_{\mathsf{eval}}(\mathsf{pp}, \mathsf{cm}, \mathbf{r}, v; f, \mathsf{st}) = 1 \quad \Longleftrightarrow \quad \mathsf{Rel}_{\mathsf{com}}(\mathsf{pp}, \mathsf{cm}, f, \mathsf{st}) = 1 \ \wedge \widetilde{f}(\mathbf{r}) = v. \tag{51}$$

The shape carried by cm determines the type of every argument in this relation. In particular, the payload C is never interpreted without its shape. A PCS must satisfy the following three properties.

Definition 3.8 (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 $\varepsilon_{\mathsf{comp}}(\lambda) = 0$.

Definition 3.9 (PCS binding). *For every PPT adversary* $\mathcal{A}$,

$$\Pr\left[\begin{array}{c} \mathsf{Eval.V}(\mathsf{pp}, \mathsf{cm}, \mathbf{r}, v, \pi) = 1 \\ \wedge\ \mathsf{Eval.V}(\mathsf{pp}, \mathsf{cm}, \mathbf{r}, v', \pi') = 1 \\ \wedge\ v \ne v' \end{array} \middle| \begin{array}{c} \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, \mu_{\max}) \\ (\mathsf{cm}, \mathbf{r}, v, \pi, v', \pi') \leftarrow \mathcal{A}(\mathsf{pp}) \end{array}\right] \le \mathit{negl}(\lambda).$$

In the experiment below, $\mathsf{Run}[\mathcal{P}^*(\mathsf{pp})]$ 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 the same execution and does not invoke an independently sampled copy of $\mathcal{P}^*$. It answers replayed oracle queries consistently, except at the challenge queries that its rewinding algorithm resamples.

Definition 3.10 (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 analyse the folding step at each recursive level in the CWSS framework of Fenzi, Moghaddas, and Nguyen [42], together with its Fiat–Shamir analysis in the random-oracle model.

Definition 3.11 (Special-sound structure [79, 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 $k_i \geq 2$, 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 3.12 (CWSS protocol [79, Definition 3]). *A public-coin interactive protocol with μ challenge rounds, where round i draws a challenge from $\mathcal{C}_i^{\ell_i}$, is* $(\ell_1, \dots, \ell_\mu)$-coordinate-wise $(k_1, \dots, k_\mu)$-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 3.13 (CWSS-to-knowledge and classical Fiat–Shamir [42, 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*

$$\kappa_{\mathsf{CWSS}} \;=\; \frac{\mu\, \ell\, (k-1)}{|\mathcal{X}|}.$$

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 with knowledge error

$$\kappa_{\mathsf{FS\text{-}CWSS}}(Q) \;=\; (Q+1) \cdot \frac{\mu\, \ell\, (k-1)}{|\mathcal{X}|},$$

where Q is the number of classical random-oracle queries made by the adversary. This statement is not a QROM theorem.

4 Setup and Commitments

We want to replace several opening claims by one claim about a shorter witness. Before constructing that fold, we must decide what each incoming commitment has already fixed. 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 opening batch: each commitment fixes its own representation and binding parameters, while the fold chooses compatible opening data for the complete batch.

We first describe the committed objects and the public schedule connecting successive folds. We then construct the shared setup and the commitment maps. With those commitments in hand, Section 5 builds one fold over an ordered list of opening groups.

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 3.7. 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 receiving fold uses the opening method prescribed by (160). Under *direct coefficient packing*, the fold retains one coefficient axis in a challenge subring and opens the mixed-field claim directly (Section 5.3). Under the established *evaluation-trace* method, for $k > 1$ the receiver first applies the tensor reduction of Section 3.7, then uses the evaluation-trace relation. Both methods produce the same ordinary recursive opening. The composition rule that assigns methods to nonterminal folds appears in Section 9.3. The terminal uses evaluation trace. When $k = 1$, its tensor reduction is the identity.

Coefficients: terminology. Fix the ring-subfield K-basis $\boldsymbol{\beta} = (\beta_0, \ldots, \beta_{k-1})$ of E from Section 3.1. The committed Boolean evaluation vector f consists of 2^μ elements of K. Extension-field values arise only after the opening contraction, and 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 shape restrictions below apply to the logical K-coordinate table, not to an element of E as a whole. These coordinates are not the monomial coefficients of $\widetilde{f}$, and the packing map can change their coefficient bounds when forming a packed ring element.

Witness shapes. The Boolean evaluation 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). The protocol currently supports the three shapes below, which suffice for most applications; the design is not limited to them, and 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 guaranteed to contain 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.

Remark 4.1 (Shape guarantees are conditional). What distinguishes the shapes operationally is *whether and how the tightened shape is validated*. The small-digit shape of recursive witnesses is validated *inside* the protocol by the digit range check (Section 6.1), itself conditional on the PCS claim being correct. The one-hot shape is validated *outside* the protocol, by the surrounding proof system, on the same committed polynomials and again conditional on the PCS claim: in the Jolt integration, booleanity and Hamming-weight sum-checks over the one-hot polynomials precede the batched opening. The external one-hot guarantee is used to sharpen the honest-response bound, not to prove Akita knowledge soundness. The Akita extractor recovers a polynomial together with the algebraic decommitment of Definition 10.21, whether or not that polynomial is one-hot. If the surrounding statement requires one-hot semantics, its own booleanity and Hamming-weight checks must establish them.

Remark 4.2 (One-hot sizing benefit). The guaranteed one-hot shape sharpens the digit sizing of the folded witness at the root fold. Each canonical source class contains only the unit coefficients permitted by its one-hot chunks. The root schedule therefore uses the fixed-weight moment-generating-function (MGF) bound of Theorem G.3, which accounts for the exact chunk, ring, challenge-support, and response-window geometry. Treating this shape only as a generic dense source with coefficient bound one is valid but loses the one-hot occupancy information.

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

$$\mathsf{shape}_{\mathsf{com}} := \big(\mathsf{Src};\ M, b, b_1; (d_A, n_A); (d_B, n_B, S_B); \mathcal{F}\big). \tag{52}$$

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, and the source coefficient constraints. Together with d_A, it 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 the corresponding common source digit depth δ_{com}, so their blocks have the same $\mathbf{A}$ input width. Stronger constraints on individual vectors do not change that width; vectors requiring different source digit depths must use separate commitment handles. The source layout also fixes the decoding map used in Definition 10.21; recording an honest padding embedding does not assert that every algebraic decommitment lies in its image. 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 $\mathit{\Pi}_B(B, S_B)$ of (67) are derived. The compression profile $\mathcal{F}$ is either empty, when the outer image is transmitted directly, or contains exactly the two triples $(\{-1, 0\}, d_{F,i}, n_{F,i})_{i\in\{1,2\}}$ for the two binary compression maps. The first map's input width follows from the complete outer image, and the second map's input width follows from the first image. Thus the profile records the two ring dimensions and ranks, while the alphabet and number of stages are fixed by the protocol.

Fields and matrix notation. We use $k \in \{1, 2, 4\}$ and the basis of Section 3.1, writing $R_{q,d} = \mathbb{Z}_q[X]/(X^d + 1)$ to keep each map's ring dimension visible. A matrix over R_{q,d_I} has n_I rows and m_I columns; its rank n_I is determined by the Module-SIS security requirement and its width m_I 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 complete public description has three layers: the opening groups entering a fold, the four fields that configure the fold itself, and the edge that connects its output to the next level. We keep these layers separate because root batching changes the groups entering the first fold, whereas setup offloading changes the opening claims passed from one 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 under that commitment at that point. We call the referenced pair of frozen shape and payload a commitment handle. Referencing the same handle twice does not run the commitment algorithm again. 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 themselves.

The root source is an ordered, nonempty list of application-supplied groups, which we write as

$$\mathcal{O}_0 = (\mathcal{A}_{0,0}, \ldots, \mathcal{A}_{0,G_0-1}). \tag{53}$$

Taking $G_0 = 1$ with one claim and no semantic padding gives the ordinary single-opening PCS interface. Akita permits application-supplied multi-group batching only at this root. After the root, the source topology is determined entirely by the preceding edge. If that edge is direct, fold $j > 0$ receives only the recursive-witness group $\mathcal{W}_j$. If the preceding fold offloaded its setup evaluation, fold j instead receives the setup-prefix group $\mathcal{S}_{j-1}$ beside that witness:

$$\mathcal{O}_j = \begin{cases} (\mathcal{W}_j), & \mathsf{edge}_{j-1} = \mathsf{Direct}, \\ (\mathcal{S}_{j-1}, \mathcal{W}_j), & \mathsf{edge}_{j-1} = \mathsf{OffloadedSetup}. \end{cases} \tag{54}$$

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

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

$$\mathsf{cfg}_j := \big(\mathsf{Open}_j, \mathsf{RingCheck}_j, \mathsf{Comp}_j, \mathsf{Norm}_j\big). \tag{55}$$

These fields select the opening method, ring-reduction route, compression profile, and response-norm certificate. They summarize the exact records of Section 5.1; they are not independently serialized choices. The absolute-level policy (160) determines Open_j, and Definition 9.2 constrains the remaining choices. Root batching belongs to the source list $\mathcal{O}_0$ and setup offloading to an edge, since both affect which claims a fold receives.

With the source and target shapes fixed, this 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

$$\mathsf{edge}_j \in \{\mathsf{Direct}, \mathsf{OffloadedSetup}, \mathsf{TerminalInnerState}\}. \tag{56}$$

A direct edge produces only $\mathcal{W}_{j+1}$. An offloaded edge produces $(\mathcal{S}_j, \mathcal{W}_{j+1})$ 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

$$\begin{aligned}\mathcal{O}_0 \xrightarrow{\mathsf{Fold}_0[\mathsf{cfg}_0;\mathsf{edge}_0]} \mathcal{O}_1 \xrightarrow{\mathsf{Fold}_1[\mathsf{cfg}_1;\mathsf{edge}_1]} \cdots \\ \xrightarrow{\mathsf{Fold}_{t-1}[\mathsf{cfg}_{t-1};\mathsf{TerminalInnerState}]} \mathcal{T}_t \xrightarrow{\mathsf{Terminal}} \mathsf{accept}.\end{aligned} \tag{57}$$

For every nonfinal edge, the target shape is exactly the source shape in (54) selected by that edge. This equality is part of the syntax; the stronger algebraic, security, and terminal conditions that make a schedule admissible are stated in Section 9.3. A schedule therefore describes one complete protocol execution, rather than an independent list of feature parameters.

Section 5 constructs the witness and transcript for all these source lists. The ordinary single-claim opening and the public batch are specializations of that construction. Section 9 supplies the producer-side setup-product stage and the compatibility conditions across that edge. Distributed proving only refines how a fixed level is executed and does not change this topology (Section 11).

Fiat–Shamir profile and transcript binding. Every schedule records a target λ_{FS} and a maximum classical Fiat–Shamir-namespace query count $Q_{\max}$. For every source group it also records the raw challenge shell, every fixed filter, a certified lower bound on $|\mathcal{C}^{\mathsf{acc}}|$ for the resulting verifier-sampled family, its pairwise-difference certificate, and the raw-draw allowance of its fixed-filter sampler. Every response-producing stage, including the terminal, records a nonce set of cardinality $N_{\mathsf{try},j}$ and a deterministic joint retry rule. The schedule digest includes these values and the complete sequence of opening shapes and level descriptions, and is absorbed before the first protocol challenge. The security claim is only for adversaries making at most $Q_{\max}$ queries to $\mathcal{H}_{\mathsf{FS}}$, counting every nonce probe. Independent queries to the setup-expansion namespace do not enter this bound.

Within a fold, each challenge coordinate is derived from a separately indexed query under a fixed round root and candidate nonce. Deriving one coordinate does not absorb another coordinate's answer. This makes the coordinate-wise forks used in Section 10 explicit in the transcript interface. An algebraic proof-of-work profile additionally binds its ordered predicate sites, targets, and nonce widths; the separate accounting appears in Section 10.

4.3 Setup

The commitment and opening algorithms use several public matrices, whose dimensions can differ across the recursion. We construct 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. To obtain a particular matrix, we take the required number of entries from the beginning of this vector and arrange them into ring elements.

Table 3: 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 (82)
cfg_j	Fold-j configuration $(\mathsf{Open}_j, \mathsf{RingCheck}_j, \mathsf{Comp}_j, \mathsf{Norm}_j)$
Src, P	Frozen source layout and its ordered polynomial count in a commitment shape
edge_j	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
$d_{\max,j}$	Largest native ring dimension among the active ring-valued rows of fold j
$\mathbf{A}$, $\mathbf{B}$, $\mathbf{D}$	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

This reuse means that the setup need only accommodate the largest matrix representation, rather than store separate entries for every matrix in every fold. The public schedule fixes how each matrix is formed, so both parties interpret the shared entries in the same way. We first identify the matrices by their roles, then define their common construction.

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. Its matrix view is the corresponding prefix interpretation of $\mathbf{S}$. For each group, the fold reuses its commitment matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{F}$. It adds one shared opening path $\mathbf{D}/\mathbf{H}$. Suppressing the group index, their shapes are:

Table 4: 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
$\vartheta_{j,c}$	Public weight that combines claim c of source group j 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
$\Delta^{\mathsf{cert}}_{f,j}$, $\Delta^{\mathsf{cert}}_{\mathsf{rsp},h,g,a}$, $\Delta^{\mathsf{cert}}_{\Sigma,h,g}$	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

– $\mathbf{A}^{(j)} \in R_{q,d_A}^{n_A \times m_A}$: inner commitment;
– $\mathbf{B}^{(j)} \in R_{q,d_B}^{n_B \times m_B}$: outer commitment, applied to each slice after schedule-fixed padding to width m_B (Section 4.5);
– $\mathbf{D}^{(j)} \in R_{q,d_D}^{n_D \times m_D}$: one unsliced partial-evaluation commitment, applied to the complete ordered partial digit vector;
– generated compression-chain matrices $\mathbf{F}_i^{(j)} \in R_{q,d_{F,i}}^{n_{F,i} \times m_{F,i}}$ for $i \in \{1, \ldots, L_F\}$, and $\mathbf{H}_i^{(j)} \in R_{q,d_{H,i}}^{n_{H,i} \times m_{H,i}}$ for $i \in \{1, \ldots, L_H\}$. Here $L_F = L_H \in \{0, 2\}$ and every map uses the alphabet $\{-1, 0\}$. The schedule fixes the common compression choice and each map's shape; see Section 4.5.

(All dimensions are per level; the level superscript is dropped when clear.)

Length and matrix entries. Fix the maximum supported variable count $\mu_{\max}$. We choose N_{setup} to accommodate every matrix required by the admitted schedules up to this size. Let the largest matrix view be

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

The maximum ranges over every matrix instance that can occur in an admitted schedule. Because the schedule family is finite and its depth is bounded, this is a finite closure calculation rather than a recursive definition at proof time. The provisioned stream length is its least power-of-two envelope

$$N_{\mathsf{setup}} \;:=\; 2^{\lceil \log_2 N_{\mathsf{view}} \rceil}. \tag{58}$$

Thus every matrix instance reads at most N_{view} coefficients. The rounding costs less than a factor of two; smaller instances read shorter prefixes of the same vector.

We now specify how these coefficients are assigned to matrix entries. 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:

$$\mathbf{M}[i,j] \;:=\; \sum_{\ell=0}^{d-1} S_{\,(im+j)\,d\,+\,\ell}\; X^{\ell} \;\in\; R_{q,d}, \qquad 0 \le i < n,\; 0 \le j < m, \tag{59}$$

so $\mathbf{M}$ uses $\mathbf{S}[0\,..\,nmd-1]$. The column number follows the same order as the committed vector (Section 5.5). The $\mathbf{B}$ view uses the polynomial-major slice input of (68). The shared $\mathbf{D}$ view concatenates the opening-digit intervals in group order, with claim, block, native subcolumn, digit, and native coefficient ordered inside each interval. Section 5.5 fixes these addresses; the last coordinate changes first. Matrices of different types or levels may therefore reuse entries of $\mathbf{S}$ while interpreting them with different values of (n, m, d).

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:

$$S_i := \mathsf{XOF}_q(\mathsf{seed};\, \mathsf{setup}, i), \qquad i = 0, 1, 2, \ldots \tag{60}$$

The public setup profile fixes the concrete function, its domain-separation labels, and its map from sampled bytes to $\mathbb{Z}_q$. We require two properties of this derivation. First, the flat-index derivation is prefix-consistent: a length-N expansion is a prefix of any longer expansion from the same seed, which is exactly what the overlapping prefix views require. Second, 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. For an allocated setup error $\delta_{\mathsf{setup}} > 0$, a sufficient draw width for this bound is

$$w \ge \lceil \log_2 N_{\mathsf{setup}} + \log_2 q + \log_2(1/\delta_{\mathsf{setup}}) \rceil\,.$$

The modulus-bit-length minimum alone does not imply this aggregate target. The setup profile must therefore record the actual sampling map and width and check the bound for the largest admitted prefix; exact rejection sampling has zero coefficient bias. 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. Domain separation makes these restrictions independent. The query bound $Q_{\max}$ used below counts only queries to $\mathcal{H}_{\mathsf{FS}}$, including every nonce probe. Queries that merely recompute public setup coefficients use $\mathcal{H}_{\mathsf{setup}}$ and do not enter the Fiat–Shamir forking loss.

4.4 Per-Matrix Ring Dimensions and Challenge Families

The matrices used by Akita need not share a ring dimension. We first explain how the dimension affects matrix costs, then give the compatibility conditions imposed by the opening method and its challenges. By Lemma 7.1, the resulting relations can use different quotient rings over the same flat coefficient vector.

Shared setup vector: overlapping prefix views (schematic). All views start at S_0 and extend to per-matrix lengths; they overlap by construction.

$$
\begin{aligned}
&\mathbf{S}: && [\,S_0\ S_1\ S_2\ \cdots && \cdots\ S_{N_{\mathsf{setup}}-1}\,]\\
&\mathbf{A}^{(j)}\ \text{view}: && [\underbrace{\bullet\ \bullet\ \cdots\ \bullet}_{n_A m_A d_A}]\\
&\mathbf{B}^{(j)}\ \text{view}: && [\underbrace{\bullet\ \cdots\ \bullet}_{n_B m_B d_B}]\\
&\mathbf{D}^{(j)}\ \text{view}: && [\underbrace{\bullet\ \cdots\cdots\ \bullet}_{n_D m_D d_D}]\\
&\mathbf{F}_i^{(j)},\ \mathbf{H}_i^{(j)}\ \text{views}: && [\ \underbrace{\bullet\ \cdots\ \bullet}_{n_{K,i} m_{K,i} d_{K,i}}\ ]
\end{aligned}
$$

Here $K \in \{F, H\}$ in the last row. Each matrix view reinterprets its prefix at its own (n, m, d) per (59); N_{setup} is the rounded provisioned envelope (58). Compression-map matrices use the same prefix rule at their scheduled ring dimensions and shapes (Section 4.5, commitment compression).

Fig. 3: Setup layout: one seed-expanded vector; the matrices $\mathbf{A}, \mathbf{B}, \mathbf{D}$ and every compression-map matrix at every level are overlapping prefix views of the same vector.

Matrix size and multiplication 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

$$m_I = \frac{L_I}{d_I}, \qquad n_I m_I d_I = \frac{T_I L_I}{d_I}, \qquad n_I m_I = \frac{T_I L_I}{d_I^2}. \tag{61}$$

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 (Sections 9.2 and 12.2).

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. Section 12.2 makes this comparison using the exact padded input widths and secure ranks. Commitment compression separately reduces the transmitted images; its two-stage binary construction and concrete sizes are given in Section 4.5 and Table 5.

Direct coefficient packing. Let $E/\mathbb{F}_q$ have degree k, and let d_{car} and h be powers of two such that $d = khd_{\mathsf{car}}$; in the protocol, d is the dimension d_A of the inner commitment matrix. We use the three rings

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

The first is the *challenge subring*, the second is the *extension-field carrier ring*, and the third is the *A ring*, over which the inner commitment matrix $\mathbf{A}$ is defined. There is a canonical embedding

$$\iota : S \longrightarrow R, \qquad Y \longmapsto X^{kh}. \tag{62}$$

Indeed, $(X^{kh})^{d_{\mathsf{car}}} = X^d = -1$ in R. The A ring is a free S-module of rank kh under this embedding:

$$R = \bigoplus_{a=0}^{kh-1} X^a \iota(S). \tag{63}$$

Thus packing retains one coefficient axis, while the other kh coefficient lanes remain explicit.

Lemma 4.3 (Challenge-subring embedding). *For every $c \in S$, the embedding (62) 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 and inserts zeros elsewhere, which proves the coefficient-norm identities. Under (63), multiplication by $\iota(c)$ is the direct sum of kh 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 a *packing fold*, that is, a fold using direct coefficient packing, the protocol samples each challenge in the challenge subring of dimension d_{car} and embeds it into the A ring. Thus d_{car}, rather than the A-ring dimension d_A, determines the family. The two dimensions may be chosen independently, subject to the divisibility condition below.

Challenge and inner-commitment dimensions. A candidate parameter set for a packing fold first selects d_{car} and an audited challenge family at that dimension. The family must meet the schedule's Fiat–Shamir entropy target, and every nonzero pairwise difference must be a unit in the challenge subring S (Section 3.2). The candidate then selects an A-ring dimension satisfying

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

The challenge norm, folded-response envelope, and resulting $\mathbf{A}$ collision radius are determined by the family at d_{car}. The A-ring dimension instead determines the embedding stride kh, the $\mathbf{A}$ module rank, its output quantisation, and its ring-arithmetic cost. In particular, 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).

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, so the challenge family does not constrain the ring dimensions of $\mathbf{B}$ and $\mathbf{D}$. We choose their dimensions by comparing the required ranks and padded sizes. Their collision bounds are the certified digit alphabet diameters (Remark 10.24), rather than the folded-response envelope, so either dimension may remain below d_A when its own rank and padding costs favor that choice.

The $\mathbf{B}$ source $\hat{\mathbf{t}}$ is a flat A-ring coefficient vector, so the selected d_B view must divide the A-ring coefficient runs that it projects. The $\mathbf{D}$ source depends on the opening method. One packing partial has kd_{car} base-field coordinates, while one evaluation-trace partial has d_A base-field coordinates. The ring views therefore require

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

Each partial occupies the corresponding number of **D** subcolumns. Multiplying by digit depth and claim–block multiplicity gives the full input width used to choose the **D** rank and the **H** compression widths. There is no protocol condition $d_D \mid d_B$.

Both $\hat{\mathbf{t}}$ and $\hat{\mathbf{e}}$ remain flat K-coefficient vectors read under schedule-fixed ring views. Their balanced-digit bounds are therefore independent of those views, and commitment binding transfers without a new decomposition (Remark 7.2). Ring switching still uses one power ladder: 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 (147); evaluation trace uses the ordinary A-ring challenge evaluation. The dimensions may vary by level and, where the grouped interface permits, by group. Each must support the required arithmetic and satisfy the security conditions for its induced commitment instance. The schedule fixes these choices, the opening method, challenge family, coordinate order, and terminal path before the first fold challenge. They cannot be reinterpreted after transcript randomness is known.

Challenge norms. The challenge family also determines how large an honest folded response can be. For a packing challenge c and an A-ring response u, the embedding of Lemma 4.3 gives

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

Thus the ℓ_1 norm ω of the selected challenge-subring family controls the deterministic response envelope and the coefficient-ℓ_∞ collision bound. Lemma 4.3 also supplies the multiplication-operator bound used by a Euclidean security analysis. Whether the schedule enables that response certificate is a separate admissibility decision.

At a packing fold, the schedule records the challenge-subring dimension, sparse shell, coefficient bounds, accepted-cardinality lower bound, raw-draw allowance, and pairwise-unit certificate. At an evaluation-trace fold, it records the corresponding full-ring challenge data. The response depth and **A** collision bound are derived for the opening method prescribed at that level; Section 12.2 explains how these choices enter parameter selection.

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). We may transmit an Ajtai image directly or compress it through another Ajtai chain before transmission. Figure 4 shows the common prefix of this pipeline and its optional compression suffix.

Blocks. We first describe one vector in the source layout and suppress its polynomial index. For an ordered tuple $(f_p)_{p\in[P]}$, we apply this block construction and the inner commitment separately to every p, obtaining $\mathbf{f}_{p,i}$, $\mathbf{s}_{p,i}$, and $\mathbf{t}_{p,i}$. The sliced outer map below then binds all these 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 of length N, constructed below. The commitment shape fixes $M := 2^{r_{\mathsf{pos}}}$ ring elements per block, so $\mathbf{f}$ occupies

$$B := \left\lceil \frac{N}{M} \right\rceil \tag{64}$$

blocks. Block $i \in [B]$ contains entries $iM, \dots, iM + M - 1$. The honest commitment algorithm canonically extends the declared ring-packed vector by zero, so block i is

$$\mathbf{f}_i[x] := \begin{cases} \mathbf{f}[iM + x] & \text{if } iM + x < N, \\ 0 & \text{otherwise,} \end{cases} \qquad x \in [M]. \tag{65}$$

The block number is represented by $r_{\mathsf{blk}} := \lceil \log_2 B \rceil$ variables in a multilinear extension. Thus block numbers $B, \dots, 2^{r_{\mathsf{blk}}} - 1$ evaluate to zero, but they do not appear in the committed vector. Since M is a power of two, an equality weight separates into a within-block factor and a block factor:

$$\widetilde{\mathsf{eq}}(\mathbf{r}, iM + x) = \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{pos}}, x)\widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{blk}}, i). \tag{66}$$

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

The zero extension in (65) specifies honest commitment formation. It is not, by itself, a predicate proved by an Akita opening. To state extraction without assuming such a predicate, 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 in a final block included in the commitment remain part of this decoded vector, even when their containing ring-element positions lie beyond the declared input length N. Write $\mathsf{Pad}_{\mathsf{shape}}(f)$ for the canonical zero extension of a declared Boolean evaluation vector before padding. 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 must separately establish membership in the image of $\mathsf{Pad}_{\mathsf{shape}}$. This distinction also applies when several natural-domain evaluation vectors are embedded into one common commitment grid.

Embedding and decomposition. First *embed* the Boolean evaluation vector into the ring: pack each run of d_A consecutive base-field coefficients (in the ring-subfield basis of Section 3.1, so extension constituents interleave canonically) into one element of R_{q,d_A}, producing the ring-packed vector $\mathbf{f}$ (Section 3.1). View $\mathbf{f}$ as blocks $(\mathbf{f}_i)_{i\in[B]} \in (R^M_{q,d_A})^B$ under (65), zero-extending only the final block. 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 instead allow 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. This bound need not be that of the original logical digit alphabet: trace packing can increase the coefficient magnitude as in (46). The schedule must account for this packed-coefficient 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:

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

That is, the inner images are digit-decomposed in balanced base b_1 (depth $\delta_{\mathsf{open}} = \lceil \log_{b_1} q \rceil$).

Sliced outer commitment ($\mathbf{B}$, at d_B). Without slicing, the width of the outer map would grow with the number of blocks. Slicing limits the matrix width by reusing one matrix on consecutive ranges of source blocks. This exchanges setup width for more relation rows. The commitment shape fixes a nonempty power-of-two slice count $S_B \le B$. It does not store a second list of boundaries. Instead, it derives the canonical block ranges

$$I_{B,s} := \left[\left\lfloor \frac{sB}{S_B} \right\rfloor, \left\lfloor \frac{(s+1)B}{S_B} \right\rfloor \right) \qquad (s \in [S_B]). \tag{67}$$

We write $\mathit{\Pi}_B(B, S_B) := (I_{B,s})_{s\in[S_B]}$ only as shorthand for these derived ranges.

The same definition covers the P ordered polynomials fixed by Src, with their common block geometry; the single-polynomial core has $P = 1$. One block of one polynomial contributes

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

elements of R_{q,d_B} to the outer map. Let $\hat{\mathbf{t}}_{p,i}^{(B)} \in R_{q,d_B}^{g_B}$ 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 $\hat{\mathbf{t}}_i$: writing $s = d_A/d_B$ and $\delta = \delta_{\mathsf{open}}$, the coefficient address changes as

$$(a\delta + u)d_A + yd_B + k \quad \longmapsto \quad ((as + y)\delta + u)d_B + k.$$

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. The $\mathbf{A}$ relation retains the original d_A view; the outer commitment uses this d_B view. Write $\hat{\mathbf{t}}$ for the corresponding concatenation in polynomial and then block order. Set

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

and form the input vector for slice s as

$$\mathbf{x}_{B,s} := \Big\|_{p\in[P]} \left(\Big\|_{i\in I_{B,s}} \hat{\mathbf{t}}_{p,i}^{(B)} \,\Big\|\, \mathbf{0}^{(L_B^{\mathsf{blk}} - |I_{B,s}|)g_B} \right) \in R_{q,d_B}^{m_B}. \tag{68}$$

Thus any shorter slice is padded inside each polynomial segment, rather than after the concatenation of all polynomials. This preserves polynomial-major order and makes the case $S_B = 1$ identical to the unsliced input.

There is one matrix $\mathbf{B} \in R_{q,d_B}^{n_B \times m_B}$. The prover reuses this same matrix for every slice and computes

$$\mathbf{u}_s := \mathbf{B}\mathbf{x}_{B,s} \in R_{q,d_B}^{n_B}.$$

The complete outer image is the single stack

$$\mathbf{u} := (\mathbf{u}_0 \| \cdots \| \mathbf{u}_{S_B-1}) =: \mathbf{B}_{\Pi_B(B,S_B)}\hat{\mathbf{t}}. \tag{69}$$

Its canonical coordinate order is slice, $\mathbf{B}$ row, and ring coefficient, with the coefficient changing first. The notation on the right denotes the logical block-diagonal map induced by (68); it does not denote S_B independently sampled matrices. The complete stack is either transmitted directly or compressed once by the $\mathbf{F}$-chain below. In particular, slicing creates neither separate commitments nor separate payloads.

This construction bounds the $\mathbf{B}$ setup view at $n_B m_B d_B$ field coefficients and sizes its Module-SIS instance at the matrix width m_B. If two certified digit vectors yield the same complete stack, they differ in some slice, and their difference in that slice is a norm-$(b^\star - 1)$ kernel vector for 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 that carries these digits. The cost is that the stacked image has $S_B\, n_B$ ring elements rather than n_B. On the compressed route, this larger image enlarges the input to the $\mathbf{F}$-chain; on the raw route, it enlarges the payload itself. It also enlarges the outer-consistency rows and the public weight evaluation. These effects enter the transition cost and successor shape of Section 12.2.

Each slice contributes n_B outer-consistency rows to the common relation. Their public weights can be combined before reading each setup coefficient, so slicing increases the weight-evaluation work without replicating the setup matrix. Appendix B.4 gives this contraction and its cost.

*Commitment compression (**F**-chain).* The outer image $\mathbf{u}$ contains all $\mathbf{B}$-slices. Sending this vector can be expensive even when the witness being folded is small. We shorten the public message by committing to a digit decomposition of $\mathbf{u}$, then applying the same operation once more to the resulting image. The intermediate image limits the second map's input width, allowing it to retain a small secure output. Both maps are sized for their complete inputs under the Module-SIS assumption. At a compressed level we use exactly two maps on each of the commitment and opening paths, $L_F = L_H = 2$; at a raw level we set $L_F = L_H = 0$ and transmit the original images. The schedule fixes this choice and every matrix shape before the commitments are formed.

Binary decomposition. 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 coefficient $x \in \mathbb{Z}_q$, take the binary expansion of the canonical representative of $-x \bmod q$ and negate its digits. This deterministic decomposition satisfies

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

Thus every coefficient is represented by δ digits, and two valid digit vectors differ coefficientwise by at most one. The signs let us certify these digits by fusing the constraint $w(w+1) = 0$ into the range-check sum-check (Remark 6.2).

Set $\widetilde{\mathbf{u}}_0 := \mathbf{u}$. At each stage $j \in \{1, 2\}$, decompose the preceding image coefficientwise. For an image with s coefficients, place digit u of coefficient v at position $v + su$, so that all coefficients' least significant digits come first, then their next digits, and so on. Pack this sequence into consecutive input ring elements of $\mathbf{F}_j$, appending zeros to fill the last element. The gadget map $\mathbf{G}_{\mathcal{A}_{F,j},\cdot}$ reverses this packing and 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; its inverse notation denotes the canonical decomposition just defined. The successive commitments are

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

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 $\hat{\mathbf{v}}_1, \hat{\mathbf{v}}_2$. We retain chain indices $j \in \{1, \ldots, L_F\}$ and $j \in \{1, \ldots, L_H\}$ below so that the same formulas cover raw transmission, when both index sets are empty.

The one public payload is

$$\mathbf{u}_{\mathsf{pub}} := \widetilde{\mathbf{u}}_{L_F} = \begin{cases} \mathbf{u}, & L_F = 0, \\ \widetilde{\mathbf{u}}_{L_F}, & L_F > 0. \end{cases} \tag{70}$$

At depth zero, this 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 latter order to $c_0, \ldots, c_{d_{F,L_F}-1}$.

The compression maps use the shared setup-prefix construction of Section 4.3. The public commitment payload is $C := \mathbf{u}_{\mathsf{pub}}$.

Choosing the 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. 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 5, each complete input image of at most 8 KiB produces a 256-byte intermediate image and a 128-byte public output. The input bound applies to the concatenation of all **B**-slice images, and separately to the complete **D**-image. Each distinct commitment has its own complete **B**-image and pair of compression maps. At opening time, we use one shared **D**-image for all groups (Section 5.4). Every compression instance in this table, including all smaller supported input widths, has at least 147 bits of estimated security in the quantum ADPS16 Core-SVP model of Appendix C. This is a per-instance estimate; the combined bound over the public family of schedules is accounted for separately in that appendix.

The small public output comes at the cost of storing and checking the two digit vectors in the recursive witness. The choice between compression and raw transmission therefore accounts for both the bytes saved now and 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. These equations become rows of the common relation matrix (Figure 6). To state them once for both commitment paths, let $\mathbf{M}$ be the original map, $(\mathbf{K}_j)_{j \in \{1,\ldots,L\}}$ its compression maps, and $\mathbf{p}$ the transmitted payload. The rows assert

$$\mathbf{M}\hat{\mathbf{x}} = \mathbf{G}_{\mathcal{A}_1,\cdot}\hat{\boldsymbol{\xi}}_1, \qquad \mathbf{K}_j\hat{\boldsymbol{\xi}}_j = \mathbf{G}_{\mathcal{A}_{j+1},\cdot}\hat{\boldsymbol{\xi}}_{j+1} \quad (1 \le j < L), \qquad \mathbf{K}_L\hat{\boldsymbol{\xi}}_L = \mathbf{p}. \tag{71}$$

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 5: 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 C).

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

$$\mathbf{M}\hat{\mathbf{x}} = \mathbf{p}. \tag{72}$$

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 is absorbed before the challenges of the fold that checks its chain. The chain therefore binds its compression digits without sending those digits 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 scans the extended witness at the common balanced alphabet, but this alone does not justify the compression maps' collision bound of one. Every compression digit also participates in the fused constraint $w(w+1) = 0$, which restricts it to $\{-1, 0\}$ (Remark 6.2). The honest decomposition is deterministic; this supplies a canonical witness, while the binary constraint bounds the openings of a cheating prover. To prove binding, we work backward from the public output until two accepted openings first disagree. The corresponding commitment equation gives a short nonzero kernel vector for that map, as formalized below.

Lemma 4.4 (Compression binding chain). *Fix a depth $L \in \{0, 2\}$, a public payload $\mathbf{p}$, and either compression rows of the form (71) when $L > 0$ or the raw row (72) 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 $\mathbf{M}_{\mathsf{phys}} := \mathbf{M}$ for an unsliced map and $\mathbf{M}_{\mathsf{phys}} := \mathbf{B}$ for the sliced outer map $\mathbf{M} = \mathbf{B}_{\Pi_B(B,S_B)}$. Write $\mathbf{M}_{\mathsf{phys}} \in R_{q,d_M}^{n_{\mathsf{phys}}\times m_{\mathsf{phys}}}$; in the sliced case, $(n_{\mathsf{phys}}, m_{\mathsf{phys}}) = (n_B, m_B)$. If the two tuples differ, then one efficiently extracts one of:*
1. a solution to the matrix instance $\mathsf{MSIS}_{q,d_M}(n_{\mathsf{phys}}, m_{\mathsf{phys}}, \eta_{\mathsf{in}})$ for $\mathbf{M}_{\mathsf{phys}}$, 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 raw 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.

Commitment pipeline (one commitment).

$$f \xrightarrow{\text{embed into } R_{q,d_A}} \text{ring-packed vector } \mathbf{f} \xrightarrow{\text{reshape}} (\mathbf{f}_i)_{i\in[B]}$$

$$\xrightarrow{\mathbf{G}_{b,M}^{-1}} (\mathbf{s}_i) \xrightarrow{\mathbf{A}} (\mathbf{t}_i) \xrightarrow{\mathbf{G}_{b_1}^{-1}} \hat{\mathbf{t}}$$

$$\hat{\mathbf{t}} \xrightarrow{\mathbf{B} \text{ on canonical slices}} \mathbf{u} \xrightarrow{\text{optional } \mathbf{F}\text{-chain}} \underbrace{\mathbf{u}_{\mathsf{pub}}}_{\text{payload}}$$

Opening-time branch (shared across the fold):

$$\text{all groups' source blocks} \xrightarrow{\text{opening-method contraction}} (e_{g,c,i})$$

$$\xrightarrow{\mathbf{G}_{b_1}^{-1} \text{ on every coordinate plane}} \hat{\mathbf{e}}$$

$$\hat{\mathbf{e}} \xrightarrow{\mathbf{D}} \mathbf{v} \xrightarrow{\text{optional } \mathbf{H}\text{-chain}} \underbrace{\mathbf{v}_{\mathsf{pub}}}_{\text{payload}}$$

Ring dimensions per arrow: $\mathbf{A}$ at d_A, $\mathbf{B}$ at d_B, $\mathbf{D}$ at d_D, the $\mathbf{F}$-chain at $(d_{F,j})_{j\in\{1,\dots,L_F\}}$, and the $\mathbf{H}$-chain at $(d_{H,j})_{j\in\{1,\dots,L_H\}}$. All chain digit vectors $(\hat{\mathbf{u}}_j)_{j\in\{1,\dots,L_F\}}$ and $(\hat{\mathbf{v}}_j)_{j\in\{1,\dots,L_H\}}$ join the next recursive witness when their chain is active. Only the payloads $\mathbf{u}_{\mathsf{pub}}, \mathbf{v}_{\mathsf{pub}}$ are transmitted; at depth zero, the corresponding payload is the raw image. With coefficient packing, the contraction is $\mathcal{L}$ and $e_i \in E[Y]/(Y^{d_{\mathsf{car}}}+1)$. With evaluation trace, the tensor reduction precedes this branch and $e_i \in R_{q,d_A}$ is the full-ring partial (93).

Fig. 4: The commitment data path and the shared opening-time branch. Each transmitted payload is either the raw Ajtai image or the terminal image of its scheduled compression chain. All hidden intermediate images are bound by relation rows (Section 7).

4.6 Commitments Formed before the Fold

To admit commitments formed before their batch is known, we fix a common certified root alphabet with $b^\star = 2^{l^\star}$ in the public commitment contract. 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 also requires $b_{z,g} \le b^\star$ for response digits. Neither base need equal the source-recomposition base b_g. Different certified alphabets would require separate range predicates; we instead derive each map's collision bound from the common digit diameter.

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

$$\eta_{B,g} = b^\star - 1. \tag{73}$$

At opening time, the group-local descriptor 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

$$\Delta_{f,g}^{\mathsf{cert}} = (b^\star - 1)\frac{b_{z,g}^{\delta_{f,g}} - 1}{b_{z,g} - 1}, \qquad \eta_{A,g} = 2\bar{\kappa}_{1,g}\Delta_{f,g}^{\mathsf{cert}}. \tag{74}$$

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 source shape remains fixed throughout this admission check. 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 want to check their opening claims together and continue with only one recursive witness. Combining the source vectors before opening them would lose their distinct points

and commitment shapes. Instead, we keep a response for each opening group and share the work that checks their combined auxiliary data.

We adopt this separation from Greyhound [81, §3.2], which keeps responses local to evaluation points and jointly authenticates their auxiliary values. In Akita, each group retains its own inner and sliced outer commitment path. We collect all partial opening evaluations under one unsliced $\mathbf{D}/\mathbf{H}$ path, then check the responses and their auxiliary values 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. The squared-norm route remains restricted to a single recursive response, as specified in Section 5.1.

Throughout this section, h denotes the fold level, g an opening group, c a claim in that group, and i the index of a source block. When explaining an operation inside one group, we suppress its group and claim indices. These local formulas supply the components of the single interaction in Figure 5.

5.1 Inputs and Parameters

We keep each claimed evaluation attached to its commitment and opening point. Fix an ordered list of G groups, each referencing a frozen commitment handle from Section 4.2. Group $g \in [G]$ contains $n_{\mathsf{clm},g}$ Boolean evaluation vectors $F_{g,c}$ under one commitment, for $c \in [n_{\mathsf{clm},g}]$, and one opening point $\mathbf{r}_g$. For local claim c, let $\pi_{g,c}$ be the public projection that selects the coordinates used by the corresponding multilinear polynomial $\widetilde{F_{g,c}}$, and let $v_{g,c}$ be the claimed value. The local claims assert

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

Here $n_{\mathsf{clm},g} = P_g$, the ordered vector count in the referenced commitment's source layout. To open the same commitment at another point, we reference that handle in another group; its source layout remains fixed. The projections allow polynomials of different multilinear dimensions to share the same instance. Each $\pi_{g,c}$ must agree with the coordinate projection induced by that vector's frozen padding embedding; the verifier checks this when admitting the opening statement. We remove duplicate opening-group requests before deriving the root opening layout. This does not remove vectors from a commitment or change its Src or P_g.

Putting local claims on one grid. The honest commitment algorithm zero-pads the Boolean evaluation 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 resulting padded vector $F^{\mathsf{pad}}_{g,c} := \mathsf{Pad}_{g,c}(F_{g,c})$ satisfies

$$\widetilde{F^{\mathsf{pad}}_{g,c}}(\mathbf{r}_g) = \phi_{g,c}\,\widetilde{F_{g,c}}(\pi_{g,c}(\mathbf{r}_g)). \tag{75}$$

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 remain in a separate natural-dimension instance; the batching layer does not recover its value from the padded vector. Consequently the heterogeneous interface below supports exactly the admitted layouts, rather than every point-and-dimension combination of the base PCS API.

The padding identity is an honest-encoding identity, not a consequence of a single accepted opening. For a malicious commitment, knowledge extraction returns a decoded full common-grid evaluation vector $F^{\star}_{g,c}$ and establishes $\widetilde{F^{\star}_{g,c}}(\mathbf{r}_g) = \phi_{g,c} v_{g,c}$. It does not establish $F^{\star}_{g,c} \in \mathrm{im}(\mathsf{Pad}_{g,c})$. An application that interprets this equation as the natural-domain claim $\widetilde{F_{g,c}}(\pi_{g,c}(\mathbf{r}_g)) = v_{g,c}$ must separately certify that image condition, or place the claim in a commitment shape with no semantic padding. This is the padding convention used by the batch knowledge-soundness theorem in Section 10.2.

The commitment shape of group g fixes its A-ring dimension $d_{A,g}$. With coefficient packing, its opening descriptor separately fixes a challenge-subring dimension $d_{\mathsf{car},g}$ and derives

$$h_{\mathsf{car},g} := \frac{d_{A,g}}{k d_{\mathsf{car},g}}, \qquad \iota_g\big(c(Y)\big) := c\big(X^{k h_{\mathsf{car},g}}\big).$$

The packing descriptor is admissible only when $k d_{\mathsf{car},g} \mid d_{A,g}$ and an audited challenge family exists at $d_{\mathsf{car},g}$. Distinct groups may use different A-ring dimensions, challenge-subring dimensions, packing factors, and

challenge families. With evaluation trace, we apply the tensor reduction to the padded vectors $(F^{\mathsf{pad}}_{g,c})_{c\in[n_{\mathsf{clm},g}]}$. The claims in a group share the point $\mathbf{r}_g$, while each claim retains its own padding factor $\phi_{g,c}$ and value $v_{g,c}$. The descriptor fixes the shared tensor-reduction basis split, the group's native tail arity, and the full-ring challenge family at $d_{A,g}$.

Let M_g be the power-of-two block length fixed by the incoming commitment, and let N_g be the number of ambient ring elements in each padded vector's ring-packed representation. The number of nonempty blocks is

$$B_g := \left\lceil \frac{N_g}{M_g} \right\rceil . \tag{76}$$

With coefficient packing, the group is admissible only if its scheduled coordinate permutation places the position and low-coefficient axes among coordinates shared by every local projection. After padding, every local claim then uses the same position and low-coefficient contraction. If no such common layout exists, the claims must be placed in separate opening instances; the batching layer does not silently move claim-specific coordinates through the group's contraction $\mathcal{L}_g$, defined in Section 5.3. Write the common point segments as $\mathbf{r}_{\mathsf{pos},g}$ and $\mathbf{r}_{\mathsf{pack},g}$. The remaining segments $\mathbf{r}_{\mathsf{tail},g,c}$ and $\mathbf{r}_{\mathsf{blk},g,c}$ retain the claim-local 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.

With evaluation trace, let $g_{g,c}$ be the packed multilinear polynomial obtained by regrouping the entries of $F^{\mathsf{pad}}_{g,c}$ in the fixed extension-field basis and interpolating the resulting Boolean evaluations. The shared tensor reduction applies to every $F^{\mathsf{pad}}_{g,c}$ under $\mathbf{r}_g$, with input value $\phi_{g,c}v_{g,c}$. Let $m_{\max}$ be the largest tail arity at the receiver and let $\rho \in E^{m_{\max}}$ be the common tensor-reduction point. To use one sum-check across different arities, we extend each smaller packed polynomial so that it does not depend on the extra variables, and we multiply its public factor by the equality polynomial fixing those variables to zero (Theorem 5.1). This extension changes only the sum-check expression; it does not pad the group's committed vector. Group g uses its native prefix ρ_g and factor θ_g. The reduction produces one scaled claim

$$h_{g,c} = \theta_g g_{g,c}(\rho_g)$$

for every $c \in [n_{\mathsf{clm},g}]$. The padding factor is already included in the tensor reduction input and is not multiplied again in the evaluation-trace scalar row. Each input point remains bound separately, and each packed claim uses the prefix and packing shape prescribed for its own group.

Write $r_{\mathsf{pos},g} := \log_2 M_g$. The coordinates $e_{g,c,i,t,u}$ below are the base-field coordinate planes of the partial evaluation defined in Equation (85). For $c \in [n_{\mathsf{clm},g}]$ and $i \in [B_g]$, 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. The last block, $i = B_g - 1$, is zero in every position past entry $N_g - 1$ of the representation. Block indices from B_g to $2^{\lceil \log_2 B_g \rceil} - 1$ are also interpreted as zero, but do not appear in the commitment. With coefficient packing, the padded claim is therefore

$$\phi_{g,c}v_{g,c} = \sum_{i\in[B_g]} \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{blk},g,c}, i) \sum_{u\in[d_{\mathsf{car},g}]} \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{tail},g,c}, u) \sum_{t\in[k]} \beta_t e_{g,c,i,t,u}. \tag{77}$$

This identity is local to group g and does not mention any other opening point. With evaluation trace, its analogue is the trace opening (131) for each packed claim $g_{g,c}(\rho_g)$.

Fold description. With the input layout fixed, we choose compatible response parameters and bind them before any challenge. Each group record contains its commitment reference, claim projections, absolute level, and the lengths and alphabets of its distinguished witness segments. A packing record adds its challenge-subring geometry; a trace record adds its tensor split and trace-packing geometry. An opening shape records these public forms without the points, claimed values, or private entries.

For an ordered incoming opening shape $\mathcal{O}_h$, we record the shared opening path and the challenge and response data for each group:

$$\begin{aligned}\mathsf{Fold} := \big(&\mathsf{OpeningMethod}; \mathsf{RingCheck}; (\mathsf{Chal}_g, b_{z,g}, \delta_{f,g})_{g\in[G]};\\ &(d_D, n_D); \mathcal{H}; \mathsf{Norm}; \boldsymbol{\delta}_{\mathsf{quot}}\big).\end{aligned} \tag{78}$$

The tag $\mathsf{OpeningMethod}$ is derived from (160). Each challenge rule Chal_g specifies the raw shell, every fixed witness-independent filter, a certified lower bound on the accepted-family cardinality, the raw-draw allowance,

the challenge family for the source blocks, and the response acceptance rule. The fold has one shared nonce set and a joint retry rule. These shared fields, including the nonce budget $N_{\mathsf{try},h}$, belong to the enclosing level description and are bound by its schedule digest. With coefficient packing, the accepted family lives in the challenge subring and acts in the A ring through (62). With evaluation trace, it lives in the full A ring. An evaluation-trace fold also records the tensor-reduction shape that converts its incoming ordinary opening before the fold challenge.

The pair (d_D, n_D) and profile $\mathcal{H}$ specify the unsliced shared opening path. As for $\mathcal{F}$ in (52), $\mathcal{H}$ is empty or contains two triples $(\{-1,0\}, d_{H,i}, n_{H,i})_{i\in\{1,2\}}$; every source path and the shared path use the same compression mode. The target commitment's profile specifies its successor's mode.

The tag RingCheck selects QuotientLift or QuotientFree. The vector $\boldsymbol{\delta}_{\mathsf{quot}}$ records the quotient digit depths in the former case and is empty in the latter (Section 7).

The response certificate Norm selects a coefficient-wise bound or an additional squared-norm certificate. Packing folds use the coefficient-wise route. For simplicity, we use the squared-norm route for a single recursive response after the setup openings have been checked and response chunking has ended (Definition 9.2). That route records the squared-norm bound, the challenge operator bound, and the exact norm-proof domain described in Section 6.2.

For each group, $b_{z,g}$ and $\delta_{f,g}$ fix the positional base and digit depth of its response. The incoming commitment's base b_g still reconstructs the source blocks; these bases need not coincide. 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 (97). Collision bounds and induced Module-SIS instances are derived from them. In the summary configuration (55), Open = OpeningMethod; RingCheck and Norm are the tags above; and Comp collects the source and target $\mathcal{F}$ profiles and the shared $\mathcal{H}$ profile.

5.2 Shared Tensor Reduction

The tensor reduction of Section 3.7 extends to several opening groups without moving their claims to a common point. Group g contains the packed polynomials $(g_{g,c})_{c\in[n_{\mathsf{clm},g}]}$ defined above. Write ℓ_g for the arity of its 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

$$m_{\max} := \max_g m_g.$$

For $w \in \{0,1\}^{m_g}$ and $z \in \{0,1\}^{m_{\max}-m_g}$, define the cylindrical extensions

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

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.

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 $(\zeta_{g,c}^{\mathsf{tensor}})$: 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

$$\sum_g \sum_{c\in[n_{\mathsf{clm},g}]} \zeta_{g,c}^{\mathsf{tensor}} \widehat{g}_{g,c}(x)\widehat{A}_{\eta,g}(x). \tag{79}$$

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 $h_{g,c} := \theta_g g_{g,c}(\rho_g)$ for every claim. The verifier checks that the sum-check final value equals $\sum_{g,c} \zeta_{g,c}^{\mathsf{tensor}} h_{g,c}$ and rejects if any θ_g is zero. The reduction binds each native input point and returns one packed claim $g_{g,c}(\rho_g)$ per input claim. An identity reduction forwards the original claims under their canonical packing identification.

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.

Proof. 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. 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

$$\text{error} \quad \frac{N_{\mathsf{tensor}} - 1}{|E|} \quad \text{and tree factor} \quad N_{\mathsf{tensor}}. \tag{80}$$

5.3 Forming the Partial Opening Evaluations

The committed multilinear polynomial is defined over $K = \mathbb{F}_q$, while its opening point lies in the extension field E. Its Boolean evaluations are base-field values, which we pack into ring elements for commitment. Opening combines them with extension-field weights; 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 an A-ring dimension d_A and a challenge-subring dimension d_{car}, which is also the dimension of the smaller ring holding each partial opening evaluation. These dimensions satisfy

$$d_A = k h_{\mathsf{car}}\, d_{\mathsf{car}}, \qquad h_{\mathsf{car}} \in \mathbb{Z}_{\geq 1}, \tag{81}$$

where $k, d_A, h_{\mathsf{car}}, d_{\mathsf{car}}$ are powers of two in the supported parameter sets. Each coefficient of the partial lies in E and needs k base-field coordinates. Thus the ratio $d_A/d_{\mathsf{car}} = kh_{\mathsf{car}}$ first accommodates the extension degree, leaving a reduction by a factor h_{car} in base-field width. This size constraint is separate from the commutation of evaluation with folding, which follows from the subring acting only on the coefficient index that has not been evaluated. Define the A ring, challenge subring, and extension-field carrier ring by

$$R := K[X]/(X^{d_A}+1), \qquad S := K[Y]/(Y^{d_{\mathsf{car}}}+1), \qquad S_E := E[Y]/(Y^{d_{\mathsf{car}}}+1). \tag{82}$$

We use the canonical embedding $\iota : S \hookrightarrow R$, $Y \mapsto X^{kh_{\mathsf{car}}}$, from (62). By Lemma 4.3, it preserves support size and coefficient norms, and multiplication has the same Euclidean operator norm in S and in the ambient ring R before reduction modulo q.

Canonical coefficient split. Use the block geometry of (64) and (65), with $i \in [B]$ and $x \in [M]$. Write the ambient element at that address as

$$F_{i,x}(X) = \sum_{j\in[d_{\mathsf{car}}]} \sum_{a\in[kh_{\mathsf{car}}]} f_{i,x,a,j} X^{a+kh_{\mathsf{car}}j}, \qquad f_{i,x,a,j} \in K. \tag{83}$$

The coefficient positions form consecutive groups of size kh_{car}: a indexes a position within group j, giving ring-coefficient index $a + kh_{\mathsf{car}}j$. The matching part of the opening point is split as $\mathbf{r}_{\mathsf{pack}} \in E^{\log(kh_{\mathsf{car}})}$ 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:

$$e_i(Y) := \sum_{x\in[M]} \sum_{a\in[kh_{\mathsf{car}}]} \sum_{j\in[d_{\mathsf{car}}]} \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{pos}}, x)\widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{pack}}, a) f_{i,x,a,j} Y^j \in S_E. \tag{84}$$

Writing

$$e_i(Y) = \sum_{j\in[d_{\mathsf{car}}]} \left(\sum_{t\in[k]} \beta_t e_{i,t,j} \right) Y^j, \qquad e_{i,t,j} \in K, \tag{85}$$

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

$$\sum_{i\in[B]} \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{blk}}, i) \sum_{j\in[d_{\mathsf{car}}]} \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{tail}}, j) \sum_{t\in[k]} \beta_t e_{i,t,j} = v. \tag{86}$$

This is one E-valued linear row. It uses neither a trace map nor a tensor-reduction sum-check. After gadget decomposition, the same row replaces $e_{i,t,j}$ by $\sum_\ell b_{\mathsf{op}}^\ell \hat{e}_{i,\ell,t,j}$ and is checked on the authenticated digit vector. Here $b_{\mathsf{op}} := b_1$ is the opening-digit base used in Section 7.2.

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:

$$\begin{aligned} c_i(Y) &\in S \quad \text{in the carrier relation,} \\ \iota(c_i) = c_i(X^{kh_{\mathsf{car}}}) &\in R \quad \text{in the inner-commitment relation.} \end{aligned} \tag{87}$$

No independent ambient challenge is sampled. If $\delta = c - c'$ is a unit in S, then $\iota(\delta)$ is a unit in R, because ι maps an inverse of δ to an inverse of $\iota(\delta)$. The special-soundness argument may therefore use the same fork in both relations.

To see why the same challenge can fold the partials and source blocks, let $\mathcal{L}$ denote the contraction in (84), so that $\mathcal{L}(\mathbf{f}_i) = e_i$. The coefficient split makes $\mathcal{L}$ linear over S:

$$\mathcal{L}\big(\iota(c)F\big) = c\,\mathcal{L}(F) \qquad \text{in } S_E. \tag{88}$$

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

$$Z_x(X) := \sum_{i\in[B]} \iota(c_i)F_{i,x}(X), \tag{89}$$

the native consistency relation is

$$\mathcal{L}(Z)(Y) = \sum_{i\in[B]} c_i(Y)e_i(Y) \qquad \text{in } S_E. \tag{90}$$

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

$$[\mathbf{A}\mathbf{G}_{b_z,m_A}\hat{\mathbf{z}}]_r = \sum_{i\in[B]} \iota(c_i)[\mathbf{G}_{b_1,n_A}\hat{\mathbf{t}}_i]_r \qquad (r \in [n_A]). \tag{91}$$

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

Base-field specialization. When $k = 1$, the extension-coordinate sum in (86) has one term. The planner may still use $d_{\mathsf{car}} < d_A$, with $h_{\mathsf{car}} = d_A/d_{\mathsf{car}}$, to shorten the partial and carrier quotient. Taking $d_{\mathsf{car}} = d_A$ and $h_{\mathsf{car}} = 1$ recovers the ordinary full-ring opening.

Evaluation Trace For evaluation trace, we first use the shared tensor reduction of Section 3.7 to obtain packed claims at their native points. We again suppress the group and claim indices to describe one partial. Let ρ be this group's packed opening point. The schedule splits this point according to the trace layout as

$$\rho = (\rho_{\mathsf{blk}}, \rho_{\mathsf{pos}}, \rho_{\mathsf{pack}}), \tag{92}$$

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

$$e_i := \langle \mathbf{a}, \mathbf{f}_i \rangle \in R_{q,d_A}, \tag{93}$$

with d_A base-field coordinates. For later use, we write the base-field width of one partial under either method as

$$d_{\mathsf{op}} := \begin{cases} kd_{\mathsf{car}} & \text{with direct coefficient packing,} \\ d_A & \text{with evaluation trace.} \end{cases} \tag{94}$$

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 $\mathbf{s}_{g,c,i}$ and $\mathbf{t}_{g,c,i} = \mathbf{A}_g\mathbf{s}_{g,c,i}$ are the already committed data, and $\hat{\mathbf{t}}_{g,c,i}$ are their base-$b_{1,g}$ digits. The slice width is the one derived in Section 4.5 under this substitution. The slice inputs $\mathbf{x}_{B,g,s}$ of (68) and 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. We can still bind them together: we concatenate their base-field digit vectors and use one opening commitment for the resulting vector. We first identify exactly which coordinates each group contributes.

At a packing level, for each local claim and source block, the prover applies the group-local contraction $\mathcal{L}_g$ of (84), using $\mathbf{r}_{\mathsf{pos},g}$ and $\mathbf{r}_{\mathsf{pack},g}$:

$$e_{g,c,i}(Y) = \sum_{u\in[d_{\mathsf{car},g}]} \left(\sum_{t\in[k]} \beta_t e_{g,c,i,t,u} \right) Y^u \in E[Y]/(Y^{d_{\mathsf{car},g}}+1). \tag{95}$$

At an evaluation-trace level, each group instead uses the full-ring partials of (93) for its tensor-reduction output, with $d_{A,g}$ base-field coordinates per partial.

The base-field layout of a packing partial is $[g,c,i,t,u]$. Its $kd_{\mathsf{car},g}$ coefficients are gadget-decomposed in base $b_{1,g}$ after splitting into native d_D-coefficient blocks (Section 5.5). Let $\hat{\mathbf{e}}_g$ be the resulting interval for group g. With evaluation trace, the layout is $[g,c,i,u]$ for $u \in [d_{A,g}]$, with the same native-block digit order. 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 in either mode. We concatenate them without padding to a common geometry.

One unsliced opening matrix commits to the whole vector. The fold selects a shared opening compression depth $L_H \in \{0,2\}$ and shared d_D and n_D such that d_D divides the partial width in base-field coordinates of every group. All outer commitments use the same mode, $L_{F,g} = L_H$ for every g. At the root these depths are two. The 8 KiB input bound of Table 5 applies separately to each complete **B**-stack and to the shared **D**-image; it is not a bound on their combined size. Define

$$d_{\mathsf{op},g} := \begin{cases} kd_{\mathsf{car},g} & \text{with coefficient packing,} \\ d_{A,g} & \text{with evaluation trace.} \end{cases} \tag{96}$$

Group g contributes

$$m_{D,g} := n_{\mathsf{clm},g} B_g \delta_{\mathsf{open},g} \frac{d_{\mathsf{op},g}}{d_D} \tag{97}$$

columns over R_{q,d_D}. Set

$$m_D := \sum_{g\in[G]} m_{D,g}, \qquad \mathbf{D} \in R_{q,d_D}^{n_D \times m_D}, \qquad \mathbf{v} := \mathbf{D}\hat{\mathbf{e}}. \tag{98}$$

The **D** columns follow the exact concatenated coefficient intervals. There is no **D** slice count, group-local **D** image, or padding to $\max_g d_{\mathsf{car},g}$. If $L_H = 0$, the one public opening payload is $\mathbf{v}_{\mathsf{pub}} = \mathbf{v}$; otherwise the one 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; we do not reuse a rank from one group or rescale the cost of a full-d_A opening.

Combining claims and forming responses. The shared opening payload binds the partials, but each group still carries its own scalar claims. We combine these claims only after binding that payload. This order prevents the prover from choosing partials that cancel under known weights. 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

$$\vartheta_{g(a),c(a)} := \zeta_{\mathsf{batch}}^a, \qquad a \in [\Lambda]. \tag{99}$$

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:

$$v_{\mathsf{root}} := \sum_{g\in[G]} \sum_{c\in[n_{\mathsf{clm},g}]} \vartheta_{g,c} \phi_{g,c} v_{g,c}. \tag{100}$$

We retain the subscript root for this batched scalar target even when the fold receives an offloaded setup opening; at an application root it is also denoted v_{batch}.

At an evaluation-trace fold, the shared tensor reduction has already bound the scaled claims $h_{g,c}$. After binding the shared opening payload, we fix the first scalar coefficient to one and sample the other $\Lambda - 1$ coefficients independently from E. These draws are separate from the claim-batching coefficients inside the tensor reduction. The trace target is $\bar{v}_{\mathsf{root}} := \sum_{g,c} \vartheta_{g,c} h_{g,c}$. When the tensor reduction is the identity, we set $\theta_g = 1$ and forward the padded input claims under their canonical packing identification. For $k = 1$, this means $\rho_g = \mathbf{r}_g$ and $h_{g,c} = \phi_{g,c} v_{g,c}$. There are no tensor-prefix challenges.

After the scalar weights are fixed, the verifier samples one mode-specific challenge $c_{g,c,i}$ for every group, local claim, and source block. A single nonce selects the complete domain-separated raw-draw tape. If every bounded fixed-filter sampler succeeds, the resulting tuple is uniform over the product of the scheduled accepted families in the random-oracle model. The prover retains the tuple only when every group response

$$\mathbf{z}_g := \sum_{c \in [n_{\mathsf{clm},g}]} \sum_{i \in [B_g]} c^A_{g,c,i} \mathbf{s}_{g,c,i} \tag{101}$$

meets that group's scheduled response certificate. A retry resamples the whole tuple, and the verifier enforces the common nonce limit.

Here $c_{g,c,i} \in \mathcal{C}_{d_{\mathsf{car},g}}$ and $c^A_{g,c,i} = \iota_g(c_{g,c,i})$ with coefficient packing. With evaluation trace, $c_{g,c,i} = c^A_{g,c,i} \in \mathcal{C}_{d_{A,g}}$ is a full-ring challenge.

Let $b_{z,g}$ and $\delta_{f,g}$ be the response base and depth fixed by group g's opening descriptor. The prover decomposes its response as

$$\hat{\mathbf{z}}_g := \mathbf{G}^{-1}_{b_{z,g}, m_{A,g}}(\mathbf{z}_g), \qquad \mathbf{z}_g = \mathbf{G}_{b_{z,g}, m_{A,g}} \hat{\mathbf{z}}_g. \tag{102}$$

The one range check need not use this positional base. It certifies every response digit in the common balanced base-$b^\star$ alphabet fixed by the public commitment contract of Section 4.6; admission therefore requires $b_{z,g} \le b^\star$. This separates the base that recomposes a response from the alphabet that bounds each of its digits.

5.5 The Witness Passed to the Next Level

We have formed the responses. To check how they were obtained, the next witness also carries the partial opening evaluations, inner images, and compression digits. Their order matters: quotient digits for the main relations precede the compression stages, and each stage carries its own quotient digits immediately after its binary digits.

First collect the response and auxiliary digits into

$$\mathbf{w}^{(h)}_{\mathsf{main}} := \Big\| _{g \in [G]} [\hat{\mathbf{z}}_g \mid \hat{\mathbf{e}}_g \mid \hat{\mathbf{t}}_g], \qquad \hat{\mathbf{c}}_a := \left(\Big\| _{g \in [G]} \hat{\mathbf{u}}_{g,a} \right) \| \hat{\mathbf{v}}_a. \tag{103}$$

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 range check certifies the common balanced base-$b^\star$ alphabet, so admission requires every carried digit base to fit that alphabet. 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. The witness is a vector of base-field coefficients. Each segment retains its own ring dimension; we do not enlarge all segments to a common ring. For one group, suppress its index. Response digits use d_A coefficients per ring element, while partial opening and inner-image digits use d_D and d_B, respectively. 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

$$\mathrm{pos}(j, y, u, k) := \big((js + y)\delta + u\big)d + k, \qquad y \in [s], \quad u \in [\delta], \quad k \in [d]. \tag{104}$$

Thus the coefficient coordinate changes first, then the digit, then the native block. For partial opening evaluations, j enumerates claim/block pairs and $(D, d) = (d_{\mathsf{op}}, d_D)$. For inner images, j enumerates claim/block/$\mathbf{A}$-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

$$\mathrm{pos}_{\hat{\mathbf{z}}}(x, u_{\mathsf{com}}, u_{\mathsf{fold}}, k) := \big(\delta_f(\delta_{\mathsf{com}} x + u_{\mathsf{com}}) + u_{\mathsf{fold}}\big) d_A + k. \tag{105}$$

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.

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 6): 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

$$\mathbf{w}^{(h+1)} := \mathbf{w}^{(h)}_{\mathsf{main}} \| \hat{\mathbf{Q}}_{\mathsf{main}} \Big\| \Big\|_{a=1}^{L_{\mathsf{comp}}} [\hat{\mathbf{c}}_a \mid \hat{\mathbf{Q}}_a]. \tag{106}$$

Every $\hat{\mathbf{Q}}$ segment is empty on the quotient-free route; at compression depth zero, the entire compression suffix is absent. 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

$$\mathrm{pos}_{\hat{\mathbf{Q}}_r}(u, t, k) := (up_r + t)d_r + k. \tag{107}$$

The range check covers these digits in the common balanced alphabet; the schedule fixes their decomposition.

Equation (106) 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 the main body and its quotient rows, segments are concatenated without padding to the largest ring dimension. The order follows the relations and compression stages, not a global sort by dimension.

Choosing the response digit depth. We choose a digit depth separately for each group's response, then certify all those digits in the common range proof. To explain the two bounds that this choice must satisfy, suppress the group index and write b for its response base $b_{z,g}$. The group stores its response as δ_f balanced base-b digit planes. Let $b^\star$ be the certified range base against which the next level checks those planes. The schedule requires $b \le b^\star$. These two bases give two different response bounds. The honest prover uses the canonical base-b decomposition. Its exact representable interval is

$$[-M_f, T_f], \qquad M_f := \frac{b}{2} \cdot \frac{b^{\delta_f} - 1}{b - 1}, \qquad T_f := \left(\frac{b}{2} - 1\right) \frac{b^{\delta_f} - 1}{b - 1}. \tag{108}$$

Thus a symmetric honest-response threshold used to choose δ_f must be at most T_f. This is the condition that makes the honest response encodable by the prescribed number of canonical digits.

The next level, by contrast, 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

$$\beta^{\mathsf{cert}}_{\mathsf{fold}} = \frac{b^\star}{2} \cdot \frac{b^{\delta_f} - 1}{b - 1}. \tag{109}$$

This second bound applies to every accepted response and determines the Module-SIS collision radius. It is independent of how the honest-response threshold used to select δ_f was obtained. When $b < b^\star$, it is strictly larger than the canonical representable interval and cannot be used to justify honest encodability.

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

$$\Delta_f^{\mathsf{cert}} = (b^\star - 1) \cdot \frac{b^{\delta_f} - 1}{b - 1}. \tag{110}$$

On the coefficient route, the schedule combines Δ_f^{cert} with the certified challenge-difference bound to derive the $\mathbf{A}$-collision radius in (74), 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.

The honest-prover analysis is kept separate from these enforced statements. Lemma G.1 and Theorem G.3 give rigorous root thresholds for dense and canonical one-hot sources when the accepted challenge distribution satisfies their hypotheses, while Appendix G.3 specifies the conditional recursive response model used by the parameter search. In every case, the selected threshold is rounded up to the least digit depth whose canonical interval (108) contains it. The wider envelope (109) is then derived from that depth for security sizing. The retry allowance $N_{\mathsf{try},h}$ is an independent schedule parameter and is accounted for by Theorem 10.9.

5.6 Interaction and Special Cases

We can now put the messages in order. The commitment handles determine the source maps; the fold record determines the shared opening path, response certificates, and ring-check route. These choices also determine the complete successor shape before any challenge. In particular, a different response certificate changes the collision bound that the incoming $\mathbf{A}$ map must support. That map's frozen rank and commitment geometry must already cover the new bound. When provisioning a new commitment, we resolve the certificate before choosing its rank, slices, and compression maps.

Transcript binding. The preamble absorbs the setup-seed identity, schedule digest, ordered commitment handles, points, projections, and claims before any challenge. The digest binds every source and target shape, coefficient order, challenge family and sampler, nonce budget, response route and integer encoding, compression path, and ring-check tag. Subsequent messages follow Figure 5. Thus the proof binds the complete ordered opening statement; a partial cannot 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 (106).

At the application root, the analyzed policy uses coefficient packing (Equation (160)). The ordered groups can refer to different commitments or to one commitment at several points. After the shared opening payload is bound, (99) combines the Λ scalar claims. Once the partials are uniquely bound, 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. The group-local commitment and opening geometries remain the ones admitted in Section 5.1.

At recursive edges, instantiate the same fold with the source list of (54). The producer-side setup check and terminal binding are given in Sections 8.2 and 9.2.

One Akita fold at level h.
Public input: the ordered groups $\mathcal{O}_h$, with their commitment handles, points, projections, and claimed values; the setup identity and schedule digest fixing the source and successor shapes, fold record, and edge.

0. Reduce the incoming claims when required. At an evaluation-trace fold with $k > 1$, absorb all tensor column partials, then draw the shared row- and claim-batching challenges. Run the one shared tensor reduction of Theorem 5.1, absorbing each message before its challenge. Bind every $h_{g,c}$ and check every $\theta_g \neq 0$. Each group retains its native prefix ρ_g. Packing and identity tensor reductions omit this step.

1. Bind all partial opening evaluations. The prover forms the mode-specific $e_{g,c,i}$, decomposes their base-field coordinates, and concatenates the group intervals. It sends the raw shared $\mathbf{D}$ image or the final $\mathbf{H}$ payload $\mathbf{v}_{\mathsf{pub}}$. Absorb that payload before any scalar-batching or fold challenge.

2. Combine the scalar claims and fold each group. Derive the weights of Section 5.4. For one claim, use weight one and no scalar-batching draw. Then use a candidate nonce to derive the complete mode-specific tuple $(c_{g,c,i})_{g,c,i}$ and form every $\mathbf{z}_g$ by (101). Reject that nonce if a bounded sampler fails or a response does not fit its scheduled certificate. A retry resamples the whole tuple; enforce the common nonce budget. Each coordinate query is separately indexed under the fixed round root and nonce, without absorbing other coordinates' answers.

3. Bind the successor witness. Assemble (106), including all quotient digits on the quotient-lift route and none on the quotient-free route. Commit using the successor's frozen shape and absorb its payload. At the final nonterminal fold, bind the canonical terminal inner state instead (Section 8.2).

4. Bind any squared-norm claims. On the eligible singleton Euclidean route, send and absorb the canonical integer encoding of $\mathcal{E}_{\mathsf{resp}}$ and, if required, the digit-inner-product claims. Their batching challenges follow these claims and the successor binding. The coefficient route sends no norm claims.

5. Check the ranges and relations. Draw α and the later batching challenges in the order specified in Sections 6.1, 6.2 and 7.4. Use the selected ring check on the rows of Figure 6; run one range pipeline on the complete witness and fuse the optional norm proof into its final stage. The fused relation sum-check authenticates those outputs against the same witness and leaves one claim $\widetilde{\mathbf{w}}^{(h+1)}(\mathbf{r}^{(h+1)}) = v^{(h+1)}$.

6. Pass the remaining claims to the next level. A direct edge produces the witness opening $\mathcal{W}_{h+1}$. An offloaded edge performs the additional setup-product stage of Section 9, then produces $(\mathcal{S}_h, \mathcal{W}_{h+1})$. The new setup group enters only the next fold. The final edge produces TerminalInnerState and no setup claim.

Fig. 5: The canonical interaction for every nonterminal fold. All groups share the opening payload, challenge retry, range pipeline, and relation check. The terminal's direct checks are specified separately in Section 8.2.

6 Norm Checks for the Recursive Witness

The binding argument assumes that the next recursive witness is small in a publicly specified sense. We certify that premise before using the witness in another 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 repays the cost of proving it. These are norm statements only: the algebraic equations that make the witness a valid fold are checked separately in Section 7.

6.1 Digit Range Check

At level $j+1$, the next commitment binds the witness vector $\mathbf{w} := \mathbf{w}^{(j+1)}$, indexed by $\{0,1\}^{\mu'}$. We write $\widetilde{\mathbf{w}} := \widetilde{\mathbf{w}}^{(j+1)}$ 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

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

for every $k = 0, \ldots, b^\star/2 - 1$. 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 k and $-(k + 1)$ has the same image $k(k + 1)$ 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

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

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

$$Q_{\mathsf{sq}}(s) := \prod_{k \in \mathcal{U}_{b^\star}} \big(s - k(k + 1)\big). \tag{111}$$

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

$$\sum_{x \in \{0,1\}^{\mu'}} \widetilde{\mathsf{eq}}(\tau_0, x) Q_{\mathsf{sq}}\big(\mathbf{s}(x)\big) = 0, \tag{112}$$

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. A Euclidean route keeps the same sum-check but uses its standard compressed message format, as specified below.

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 $\mathbf{s}(x) - k(k+1)$ for $k \in \mathcal{U}_{b^\star}$, 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 $\mathbf{s}(x) - k(k+1)$ 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 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 does not change this range tree. The separate offloading step is described in Section 9.1.

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 Equation (111), since the two roots k and $-(k+1)$ 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 part 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 the carried-claim coefficient γ and the distinct binariness coefficient ζ_{bin}. Their claims are combined in the canonical fused sum-check (158). 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 $\mathcal{I}_{\mathsf{bin}}$ is the union of the schedule-known witness spans occupied by compression digits, including their commitment parts. Its restricted equality weight can therefore be evaluated without enumerating the padded witness; Appendix B.2 gives the exact verifier calculation.

6.2 Certifying the ℓ_2 Response Norm

The digit range check bounds each digit of the recursive witness. For the folded response, these bounds also give a bound on each reconstructed coefficient, but they allow every coefficient to be large at once. When the response has a smaller total squared norm, we can use that stronger bound to obtain tighter Module-SIS parameters for the inner commitment. We now show how to certify it.

The quantity we need is the norm of the response represented by the digits. A digit in position h contributes b^h times its value to the response coefficient, where b is the response base. We must add these contributions before squaring: the square of their sum includes products between different digits that would be missing if we squared each digit separately. We therefore reconstruct each response coefficient as an integer and sum the squares of these reconstructed values. The digit range check remains necessary, since the other parts of the recursive witness still rely on their coefficient bounds.

The response norm statement. Let m_A be the input width of $\mathbf{A}$ and let d_A be its ring dimension. The reconstructed response $\mathbf{z}$ therefore has $N_A := m_A d_A$ base-field coordinates. These are the coefficients of $\mathbf{z}$, not its stored digits $\hat{\mathbf{z}}$. We index them by $u \in [N_A]$. To locate the digits of each response coefficient in the committed witness, we use the injective map

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

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

$$z_u^{\mathbb{Z}} := \sum_{h=0}^{\delta_f - 1} b^h \mathbf{w}^{(j+1)}\big(\pi(u, h)\big). \tag{113}$$

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

$$\mathcal{E}_{\mathsf{int}}(\hat{\mathbf{z}}) := \sum_{u=0}^{N_A - 1} \big(z_u^{\mathbb{Z}}\big)^2 \quad \text{satisfies} \quad \|\mathbf{z}\|_{2,\mathrm{coef}}^2 \le \mathcal{E}_{\mathsf{int}}(\hat{\mathbf{z}}). \tag{114}$$

The Euclidean route requires the exact statement

$$\mathcal{E}_{\mathsf{int}}(\hat{\mathbf{z}}) = \mathcal{E}_{\mathsf{resp}} \qquad \text{and} \qquad \mathcal{E}_{\mathsf{resp}} \le S_{\max}, \tag{115}$$

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 (114) 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 establishes an equality in $\mathbb{F}_q$, whereas we need an equality of integer squared norms. We can use the field equality directly when both integers lie in $[0, q)$, so reduction modulo q loses no information. To check this condition before the proof starts, we bound the largest squared norm allowed by the certified digit ranges. For each response plane h, let $B_{\mathsf{dig},h}$ be the largest absolute integer admitted by its scheduled digit alphabet, and define

$$U_{\mathsf{dir}} := N_A \left(\sum_{h=0}^{\delta_f - 1} b^h B_{\mathsf{dig},h} \right)^2 . \tag{116}$$

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

$$\sum_{x \in \{0,1\}^{\mu'}} z_{\mathsf{int}}(x)^2 = \mathcal{E}_{\mathsf{resp}}. \tag{117}$$

Both sides then represent integers below q, so the accepted field identity determines their integer values uniquely.

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 (117), 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.

Proof. 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 $\mathbb{Z}$.

Digit-expanded norm reconstruction. When the complete squared norm can exceed q, one field identity no longer determines its integer value. We instead expand the squared norm into inner products between digit planes, where plane h collects digit h of every response coefficient. We split each inner product into short segments so that its integer value lies in $(-q/2, q/2)$ and can be recovered uniquely from its residue. The verifier then reconstructs the complete norm using integer arithmetic. Write the Boolean digit vectors as $z_h(x) := \mathbf{w}^{(j+1)}(\pi(x,h))$ for $h \in [\delta_f]$, with zero padding outside $[N_A]$. Partition $[N_A]$ into public consecutive segments I_t. For every segment and every $0 \le h \le k < \delta_f$, the prover supplies the claim

$$p_{t,h,k} := \sum_{u \in I_t} z_h(u) z_k(u) \in \mathbb{F}_q. \tag{118}$$

The schedule must choose the partition so that, for every (t, h, k),

$$|I_t| B_{\mathsf{dig},h} B_{\mathsf{dig},k} < q/2. \tag{119}$$

The verifier takes the unique centered integer lift $\bar{p}_{t,h,k} \in (-q/2, q/2)$ and reconstructs

$$\mathcal{E}_{\mathsf{resp}} = \sum_t \left(\sum_{h=0}^{\delta_f - 1} b^{2h} \bar{p}_{t,h,h} + 2 \sum_{0 \le h < k < \delta_f} b^{h+k} \bar{p}_{t,h,k} \right). \tag{120}$$

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 (119). If (118) holds in $\mathbb{F}_q$, then $\bar{p}_{t,h,k}$ is the integer inner product $\sum_{u\in I_t} z_h(u)z_k(u)$.*

Proof. The triangle inequality and the digit bounds give

$$\left|\sum_{u\in I_t} z_h(u)z_k(u)\right| \le |I_t|B_{\mathsf{dig},h}B_{\mathsf{dig},k} < q/2.$$

There is exactly one integer in $(-q/2, q/2)$ with the claimed residue.

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 (120) 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 (115).*

Proof. For each segment I_t, expand $\sum_{u\in I_t}(\sum_h b^h z_h(u))^2$ and separate the diagonal and off-diagonal digit-plane pairs. Summing the resulting identity over the partition of $[N_A]$ gives (120).

Fusion with the final range-tree sum-check. The norm proof shares the last sum-check of the range tree. 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 $\mathsf{idx}(t,h,k)$ be the position of the triple (t,h,k) in a public canonical ordering. The number of triples is

$$N_{\mathsf{pair}} := |\{(t,h,k) : 0 \le h \le k < \delta_f\}| = |\{I_t\}|\,\frac{\delta_f(\delta_f+1)}{2}. \tag{121}$$

Using this ordering, the verifier sets

$$P_{\mathsf{norm}}(x) := \sum_{t,h\le k} \zeta_{\mathsf{pair}}^{\mathsf{idx}(t,h,k)} \mathbf{1}_{I_t}(x) z_h(x) z_k(x),$$
$$C_{\mathsf{norm}} := \sum_{t,h\le k} \zeta_{\mathsf{pair}}^{\mathsf{idx}(t,h,k)} p_{t,h,k}.$$

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

$$\sum_{x\in\{0,1\}^{\mu'}} (P_{\mathsf{rng}}(x) + \zeta_{\mathsf{norm}} P_{\mathsf{norm}}(x)) = C_{\mathsf{rng}} + \zeta_{\mathsf{norm}} C_{\mathsf{norm}}. \tag{122}$$

The direct norm term has degree two. Each digit-pair term has degree three after its segment indicator is included. The norm term does not carry the range summand's common equality factor, so the fused stage uses the standard compressed message format. Fusion adds no sum-check rounds. If the last range product has degree $d_i \in \{2,4\}$, its summand with the equality factor has degree d_i+1, which is at least the degree of either norm term. The standard round message therefore sends d_i+1 field elements, one more than the equality-factored range message alone (Table 6).

Binding the norm values to the committed witness. At the final range-tree sum-check point $\mathbf{r}$, the direct proof leaves the evaluation $\widetilde{z_{\mathsf{int}}}(\mathbf{r})$. The digit-expanded proof leaves the digit evaluations $\widetilde{z_h}(\mathbf{r})$ for $h \in [\delta_f]$. 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

$$\eta_{\mathsf{norm}}\widetilde{z_{\mathsf{int}}}(\mathbf{r}) = \eta_{\mathsf{norm}} \sum_{u\in[N_A]} \widetilde{\mathsf{eq}}(\mathbf{r},u) \sum_{h=0}^{\delta_f-1} b^h \mathbf{w}^{(j+1)}\big(\pi(u,h)\big), \tag{123}$$

$$\sum_{h=0}^{\delta_f-1} \eta_{\mathsf{norm}}^{h+1}\widetilde{z_h}(\mathbf{r}) = \sum_{u\in[N_A]} \sum_{h=0}^{\delta_f-1} \eta_{\mathsf{norm}}^{h+1}\widetilde{\mathsf{eq}}(\mathbf{r},u)\mathbf{w}^{(j+1)}\big(\pi(u,h)\big) \tag{124}$$

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 (123) in the direct case or of (124) 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

$$\sum_{x} \left(P_{\mathsf{base}}(x) + P_{\mathsf{bind}}(x)\right) = C_{\mathsf{base}} + C_{\mathsf{bind}}. \tag{125}$$

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 $\widetilde{\mathbf{w}}^{(j+1)}$ already carried by the recursive protocol.

The terminal response. At the evaluation-trace terminal there is no next witness and hence no recursive norm proof: the revealed response and its scheduled coefficient-wise or squared-norm bound are checked directly as specified in Section 8.2.

Security interface. The checks above certify (115); the algebraic obligations remain those of Section 7. The soundness error of the two norm protocols is accounted for in Lemma 10.34, and the weak-binding reduction turns this certified premise into the route-specific $\mathbf{A}$ collision radius in Corollary 10.35. Honest-prover bounds used to select $S_{\max}$ are stated in Section 10.1; they are not part of the norm-check soundness argument.

7 Checking the Ring Relations

The norm checks of Section 6 constrain the recursive witness to the alphabets and, when selected, the Euclidean bound assumed by the binding argument. They do not establish that this witness is the result of a valid fold. We now check the fold, commitment, and evaluation equations on those same witness digits.

The claimed evaluations give scalar equations over E, while the fold and commitment equations hold over the commitment rings. After reducing the ring equations to field equations, we combine both kinds of constraints with the range-check output in one sum-check. Figure 6 gives the complete set of rows.

7.1 Opening-Method Evaluation Rows

We begin with one evaluation claim and suppress the group and claim indices. Its weight on each opening digit depends on the selected opening method: direct coefficient packing or evaluation trace. We derive the two forms separately, then combine the rows for all claims.

Direct coefficient packing. We expand the scalar equation of (86) on the committed opening digits. For one claim, the index $i \in [B]$ selects one of the B block partials. Within e_i, the index $j \in [d_{\mathsf{car}}]$ selects a carrier coefficient, $t \in [k]$ selects its coordinate in the extension basis $(\beta_t)_{t\in[k]}$, and $\ell \in [\delta_{\mathsf{open}}]$ selects a digit. Thus each partial is reconstructed as

$$e_i(Y) = \sum_{j\in[d_{\mathsf{car}}]} \left(\sum_{t\in[k]} \beta_t \sum_{\ell\in[\delta_{\mathsf{open}}]} b_{\mathsf{op}}^{\ell} \hat{e}_{i,\ell,t,j} \right) Y^j, \tag{126}$$

where $b_{\mathsf{op}} = b_1$ is the opening base and $\hat{e}_{i,\ell,t,j} \in K$. A digit in position ℓ contributes b_{op}^{ℓ} times its value to its basis coordinate. Substituting this expansion into (86) gives the weight of each digit in the claimed evaluation:

$$\sum_{\substack{i\in[B],\ \ell\in[\delta_{\mathsf{open}}]\\ t\in[k],\ j\in[d_{\mathsf{car}}]}} \underbrace{\widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{blk}}, i)\, b_{\mathsf{op}}^{\ell}\, \beta_t\, \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{tail}}, j)}_{\omega_{\mathsf{dir}}(i,\ell,t,j)} \hat{e}_{i,\ell,t,j} = v. \tag{127}$$

Equivalently, $\boldsymbol{\omega}_{\mathsf{dir}}^{\top}\hat{\mathbf{e}} = v$. The fixed basis elements β_t combine all k coordinate planes into one E-valued row, without a random projection. This scalar equation is checked directly over E. The packing-consistency

equation (90) remains a separate ring-valued row; on the quotient-lift route, that row receives a challenge-subring quotient. The scalar row needs neither a quotient nor an α-weight.

When $k = 1$, the basis sum has one term, but the row remains (127); the challenge subring may still have $d_{\mathsf{car}} < d_A$.

The verifier evaluates the public weight of this row at the final sum-check point using its factored coefficients and opening-digit addresses in the witness. Appendix B.2 gives the address formula and its evaluation without constructing the full row.

Evaluation-trace row. For a nonterminal fold using evaluation trace, the tensor-reduction prefix first fixes a reduced extension-field claim $g(\rho) = \bar{v}$. The full-ring embedding packs the tail coordinates of g into the partials $e_i \in R_{q,d_A}$ of (93). We want a linear functional on these ring elements that recovers the evaluation at ρ.

The within-block coordinates ρ_{pos} have already been used to form each partial e_i. Of the remaining coordinates in (92), ρ_{blk} selects the block and ρ_{pack} evaluates the coordinates packed into its ring element. The block coordinates assign the following weight to block i:

$$\chi_{\mathsf{blk}}(i) := \widetilde{\mathsf{eq}}(\rho_{\mathsf{blk}}, i), \qquad i \in [B]. \tag{128}$$

For the coordinates within each ring element, we pack the equality weights using the trace-packing map $\psi : E^{d_A/k} \to R_{q,d_A}$. This gives

$$\check{\chi}_{\mathsf{pack}} := \psi\Big(\big(\widetilde{\mathsf{eq}}(\rho_{\mathsf{pack}}, u)\big)_{u \in [d_A/k]}\Big). \tag{129}$$

Taking a trace against this packed vector recovers the corresponding weighted sum of extension-field coordinates. We use the public linear functional

$$\mathcal{T}_{\rho_{\mathsf{pack}}}(Z) := \frac{k}{d_A} \operatorname{Tr}_H(Z\, \sigma_{-1}(\check{\chi}_{\mathsf{pack}})). \tag{130}$$

The trace identity of Hachi [79, Theorem 2] turns the packed opening into the field row

$$\sum_{i \in [B]} \chi_{\mathsf{blk}}(i)\, \mathcal{T}_{\rho_{\mathsf{pack}}}(e_i) = \bar{v}. \tag{131}$$

To express this row on the recursive witness, expand each coefficient of e_i in base b_{op}. The index $\nu \in [d_A]$ now selects a full-ring coefficient, and $\ell \in [\delta_{\mathsf{open}}]$ selects its digit. By linearity of the trace functional, the weight on opening cell (i, ℓ, ν) is

$$\omega_{\mathsf{Tr}}(i, \ell, \nu) := \chi_{\mathsf{blk}}(i) b_{\mathsf{op}}^{\ell} \mathcal{T}_{\rho_{\mathsf{pack}}}(X^{\nu}). \tag{132}$$

The verifier evaluates the multilinear extension of these weights using the same offset-equality method as for the direct row.

In this local calculation, when the final tensor-reduction check is fused with this relation, we multiply both sides of (131) by the transparent factor $A_\eta(\rho)$, the value at ρ of the public tensor-reduction accumulator defined in Section 3.7. The tensor-reduction verifier first checks that this factor is nonzero. The fused row therefore authenticates the forwarded value and requires no inversion. In the shared reduction, the corresponding factor is θ_g: it also includes the equality factor on any extra variables introduced for the common sum-check. The identity reduction instead uses $\theta_g = 1$, as specified in Section 5.1.

One scalar row for the batch. We now restore the group and claim indices. For packing, multiply each local digit weight by $\vartheta_{g,c}$ and compare the sum with the target already fixed in (100):

$$\sum_{g,c} \vartheta_{g,c} \sum_{i \in [B_g]} \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{blk},g,c}, i) \sum_{u \in [d_{\mathsf{car},g}]} \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{tail},g,c}, u) \sum_{t \in [k]} \beta_t \sum_{\ell} b_{1,g}^{\ell} \hat{e}_{g,c,i,\ell,t,u} = v_{\mathsf{root}}. \tag{133}$$

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

With evaluation trace, the corresponding target and scalar row are

$$\bar{v}_{\mathsf{root}} := \sum_{g \in [G]} \sum_{c \in [n_{\mathsf{clm},g}]} \vartheta_{g,c} h_{g,c}, \qquad \sum_{g \in [G]} \sum_{c \in [n_{\mathsf{clm},g}]} \vartheta_{g,c} \theta_g \boldsymbol{\omega}_{\mathsf{Tr},g}^{\top} \hat{\mathbf{e}}_{g,c} = \bar{v}_{\mathsf{root}}, \tag{134}$$

where $\boldsymbol{\omega}_{\mathsf{Tr},g}$ is derived from the group-local point ρ_g and evaluation-trace packing shape. Thus batching these claims also imposes no common opening point.

Binding and soundness. For either method, the complete $\mathbf{D}/\mathbf{H}$ payload binds the opening digits $\hat{\mathbf{e}}$ before the fold challenges are sampled. The next recursive commitment authenticates the final evaluation of this same digit vector. At an evaluation-trace fold, the tensor-reduction prefix has already fixed each scaled claim $h_{g,c} = \theta_g g_{g,c}(\rho_g)$ before this opening payload is bound; at a packing fold, the scalar claim is the original opening claim.

The selected scalar equation is then an ordinary row of the relation sum-check. Row batching first combines the relation rows into one claim. Conditioned on a nonzero residual in that claim, combining it with the range claim under the later challenge γ causes the residuals to cancel with probability at most $1/|E|$. The fused sum-check then checks the combined claim, as described in Section 7.4. This batching error is separate from the error of reducing the ring identities to field identities and from the coordinate-wise forking loss. For evaluation trace, the tensor-reduction error is also accounted for separately; multiplication by each group's checked nonzero factor θ_g preserves its scalar equation.

7.2 The Common Ring Relation

The scalar row checks the claimed evaluations, but we must still show that the partials and responses come from the incoming commitments. We enforce that connection group by group. Each group contributes its fold-evaluation and inner-consistency rows and its existing sliced outer path. Only the opening path and scalar row span all groups.

These rows need not share a ring dimension. Each uses the dimension and coefficient field of its own map; the opening digits and quotient digits retain the corresponding native coefficient blocks. Thus the same batch can contain distinct $\mathbf{A}_g$, $\mathbf{B}_g$, and shared $\mathbf{D}$ dimensions, as well as smaller rings for successive compression maps. The public schedule fixes the dimensions and compatible coefficient projections; batching does not require padding every row to the largest ring.

Recompose $\mathbf{z}_g$ by (102), 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 raw image $\mathbf{u}_g$ or $\mathbf{v}$; for a raw path, that image is already the public payload. The equations in Figure 6 are therefore linear in the complete witness of (106) 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

We have expressed the correctness of a fold as a collection of ring equations. To check them by field sum-check, we must account for multiplication modulo $X^d + 1$. Simply substituting a field challenge for X is insufficient: an equality modulo $X^d + 1$ need not remain an equality after that substitution.

We use two ways to account for this reduction. The first introduces a quotient polynomial that turns the ring equation into an ordinary polynomial identity, which we can then evaluate at a field challenge. The second incorporates reduction modulo $X^d + 1$ directly into the verifier's linear weights. The quotient method adds private coefficients that must be carried through the recursion; the direct method avoids those coefficients but changes the work needed to evaluate the weights. We first express the ring rows in a common form, then derive both checks from that form.

A Common Normal Form for the Ring Rows Before deriving either check, we isolate the structure shared by the ring rows. Each row multiplies public ring elements by polynomials whose coefficients are stored in the recursive witness, then compares their sum with a public target. Both checking methods will use these same coefficients and public multipliers.

To write the rows uniformly, we must account for their different ring degrees and for where they read the witness. In particular, the coefficients of one polynomial need not occupy consecutive positions in the full witness. We record these positions explicitly so that the checking rule can recover each polynomial from the existing witness layout.

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:

$$\sum_{c,i} c_{g,c,i}(Y)e_{g,c,i}(Y) = \mathcal{L}_g(\mathbf{G}_{b_g,M_g}\mathbf{z}_g)(Y) \quad \text{in } E[Y]/(Y^{d_{\mathsf{car},g}}+1). \tag{135}$$

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

$$\sum_{c,i} c_{g,c,i}(X)e_{g,c,i}(X) = \mathbf{a}_g^{\top}\mathbf{G}_{b_g,M_g}\mathbf{z}_g(X) \quad \text{in } R_{q,d_{A,g}}. \tag{136}$$

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.

$$\sum_{c,i} c^{A}_{g,c,i}\mathbf{t}_{g,c,i} = \mathbf{A}_g\mathbf{z}_g \quad \text{in } R_{q,d_{A,g}}^{n_{A,g}}. \tag{137}$$

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.

$$\mathbf{B}_g\mathbf{x}_{B,g,s} = \mathbf{u}_{g,s}, \qquad g\in[G], \quad s\in[S_{B,g}]. \tag{138}$$

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.

$$\mathbf{D}\,\mathsf{concat}(\hat{\mathbf{e}}_0,\dots,\hat{\mathbf{e}}_{G-1}) = \mathbf{v}. \tag{139}$$

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 (71) 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 (133) for coefficient packing or (134) 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 (103) and (106).

Fig. 6: 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.

Let $\mathsf{Rows}_{\mathsf{ring}}$ be the ordered set of active ring-valued rows in Figure 6. 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

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

For the ordinary $\mathbf{A}, \mathbf{B}, \mathbf{D}, \mathbf{F}$, and $\mathbf{H}$ rows, $K_r = K$, where $K = \mathbb{F}_q$ is the base field fixed in Section 4.1. The carrier row of coefficient packing instead has $K_r = E$ and is represented by its fixed $\mathbb{F}_q$-basis planes. The public schedule fixes d_r, the row's position in the canonical row order, and every witness interval read by the row. We use

$$d_{\max} := \max_{r \in \mathsf{Rows}_{\mathsf{ring}}} d_r \tag{140}$$

for the largest native dimension among these active rows.

We describe those reads at the coefficient level. Let $N_{\mathsf{w}} := |\mathbf{w}|$ and write $\mathbf{w} = (w_u)_{u \in [N_{\mathsf{w}}]}$ for the outgoing witness of (106), before its zero padding to length $2^{\mu'}$. We use the canonical embedding of its base-field entries into K_r. A row can read several polynomials from this witness, and the same witness coordinate can contribute to more than one such polynomial. We index these polynomial occurrences by a finite set Occ_r. For occurrence $c \in \mathsf{Occ}_r$, the address map specifies which witness coordinate supplies its coefficient of X^j:

$$\mathsf{addr}_{r,c} : [d_r] \longrightarrow [N_{\mathsf{w}}] \tag{141}$$

The map includes the scheduled subcolumn, digit, and compression-layer coordinates. Reading the witness at these addresses gives the polynomial

$$W_{r,c}(X) := \sum_{j=0}^{d_r-1} w_{\mathsf{addr}_{r,c}(j)} X^j \in R_r. \tag{142}$$

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

$$Z_r(X) := \sum_{c \in \mathsf{Occ}_r} A_{r,c}(X) \circledast_{d_r} W_{r,c}(X) - Y_r(X) = 0 \quad \text{in } R_r, \tag{143}$$

where $Y_r \in R_r$ is public and $\circledast_d$ denotes negacyclic multiplication modulo $X^d + 1$. We now identify the native dimensions for the rows of Figure 6. Group g's fold-evaluation row uses $d_{\mathsf{car},g}$ with coefficient packing and $d_{A,g}$ with evaluation trace. Its inner-consistency rows use $d_{A,g}$ and its sliced outer rows use $d_{B,g}$; the opening-consistency rows use d_D; and each compression row uses the dimension of its own $\mathbf{F}_j$ or $\mathbf{H}_j$ map. In a matrix row, the setup entries supply some of the coefficients of $A_{r,c}$, while the opposite side is represented by multipliers for gadget recomposition. The mode-specific scalar opening row is not in $\mathsf{Rows}_{\mathsf{ring}}$: it is already an E-linear row and joins the field relation only after the ring-valued rows have been reduced.

Quotient Lifting We first turn each ring equation into an ordinary polynomial identity. In (143), choose the unique degree-$< d_r$ representative of every element of R_r. If the row is valid modulo $X^{d_r} + 1$, the difference between its two sides is divisible by this polynomial. The prover supplies the quotient $Q_r \in K_r[X]$, of degree less than d_r, so that we can check the identity

$$\sum_{c \in \mathsf{Occ}_r} A_{r,c}(X) W_{r,c}(X) - Y_r(X) = (X^{d_r} + 1) Q_r(X), \tag{144}$$

in $K_r[X]$. Each product on the left has degree at most $2d_r - 2$, so a valid row admits a quotient within the stated degree bound.

Once these polynomials are fixed, the verifier can test the identity at a random point $\alpha \leftarrow E$. This determines the transcript order: the folded witness and every quotient are bound by the next commitment before α is sampled. Evaluating at $X = \alpha$ gives

$$\sum_{c \in \mathsf{Occ}_r} \sum_{j=0}^{d_r-1} A_{r,c}(\alpha) \alpha^j w_{\mathsf{addr}_{r,c}(j)} - Y_r(\alpha) - (\alpha^{d_r} + 1) Q_r(\alpha) = 0. \tag{145}$$

The witness weight in this equation factors into the row-dependent value $A_{r,c}(\alpha)$ and the universal coefficient weight α^j. The price of this factorization is the private polynomial Q_r, whose digits belong to the next recursive witness.

To see how the two opening methods differ, fix one group and suppress its index. In the displays below, i ranges over its claim–block pairs; for a single claim this is simply $i \in [B]$. The contraction is common to all claims in the group by Section 5.1. With coefficient packing, the fold-evaluation row is over the carrier ring of degree d_{car} with coefficient field E. We express its quotient in the fixed basis $\boldsymbol{\beta}$ as

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

Its evaluated equation is

$$\sum_i c_i(\alpha) e_i(\alpha) - \mathcal{L}(\mathbf{G}_{b,M}\mathbf{z})(\alpha) - (\alpha^{d_{\mathsf{car}}}+1) Q_{\mathsf{car}}(\alpha) = 0. \tag{146}$$

The k base-field planes $(q_{t,j})$ represent one polynomial over E. Its coefficient powers run through $\alpha^{d_{\mathsf{car}}-1}$, and its modulus contributes $\alpha^{d_{\mathsf{car}}}+1$. Combining the planes does not change this modulus to $X^{kd_{\mathsf{car}}}+1$.

The inner-consistency rows use a different ring: the challenge enters the $\mathbf{A}$ ring through the embedding $Y \mapsto X^{kh}$. Evaluating the embedded polynomial at $X = \alpha$ therefore gives $c_i(\alpha^{kh})$. For an $\mathbf{A}$ row $r \in [n_A]$, we obtain

$$[\mathbf{A}(\alpha)\mathbf{z}(\alpha)]_r - \sum_i c_i(\alpha^{kh})[\mathbf{t}_i]_r(\alpha) - (\alpha^{d_A}+1) Q_{A,r}(\alpha) = 0. \tag{147}$$

Here $\mathbf{A}(\alpha)$ and $\mathbf{z}(\alpha)$ evaluate the degree-$< d_A$ polynomial representatives entrywise. This accounts for the two evaluations of the same challenge polynomial: $c_i(\alpha)$ in the carrier equation and $c_i(\alpha^{kh})$ in the inner-consistency equations. These expressions coincide when $kh = 1$; in general, the two equations require their respective evaluations.

With evaluation trace, the challenges already lie in the full ring of degree d_A, so both fold rows use $c_i(\alpha)$. The fold-evaluation row becomes

$$\sum_i c_i(\alpha) e_i(\alpha) - \mathbf{a}(\alpha)^\top \mathbf{G}_{b,M}\mathbf{z}(\alpha) - (\alpha^{d_A}+1) Q_{\mathsf{eval}}(\alpha) = 0, \tag{148}$$

and the $\mathbf{A}$ rows use $c_i(\alpha)$ in place of $c_i(\alpha^{kh})$.

With either opening method, the $\mathbf{B}, \mathbf{D}, \mathbf{F}$, and $\mathbf{H}$ families use the ordinary lift at their scheduled dimensions, with modulus factor $\alpha^d + 1$. The active scalar opening row, given by (131) in the evaluation-trace case, joins the field relation unchanged and has no quotient or α-weight.

Soundness for mixed ring dimensions. The row families above use rings of different dimensions and share the single challenge α. The following lemma states the resulting guarantee for a generic relation whose row blocks have their own ring dimensions. Fix ring dimensions $d_1, \ldots, d_K$ (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 $b = 1, \ldots, K$, 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 $\alpha \in \mathbb{F}_{q^k}$. Define the per-block evaluated identity

$$\mathsf{M}^{(b)}(\alpha)\,\mathbf{w}^{(b)}(\alpha) = \mathbf{h}^{(b)}(\alpha) + (\alpha^{d_b}+1)\,\mathbf{r}^{(b)}(\alpha), \tag{149}$$

where, as in (50), 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 3.6 (computed blockwise, with modulus $X^{d_b}+1$) satisfy (149) for every α.*

2. (Soundness.) *If some block relation fails, then for any choice of quotients the identities* (149) *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* $\alpha \leftarrow \mathbb{F}_{q^k}$ *with probability at most* $(2\,d_{\max}-1)/q^k$.
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.*

Proof. (1) is Lemma 3.6 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 (149) 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 (149) 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 (49) restricted to the first $\log d_b$ variables.

One sum-check for all ring dimensions. Append all quotient coefficients to the committed coefficient vector before sampling α. For block b, use the augmented matrix $[\mathsf{M}^{(b)}(\alpha) \mid -(\alpha^{d_b}+1)I]$, placed in the prescribed witness and quotient slices with zeros elsewhere. Standard random-coefficient batching of these per-block sums (Section 3.4) checks all blocks in one sum-check over the common augmented witness multilinear polynomial.

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.

By definition, $d_{\max}$ in (140) already accounts for the opening method and every active compression row at its own dimension. The mixed-dimension argument of Lemma 7.1 bounds the probability that any false lifted relation passes the evaluation check by $(2d_{\max}-1)/|E|$. With coefficient packing, a false subring identity contributes at most $(2d_{\mathsf{car}}-1)/|E|$. Combining the k coordinate planes in the fixed basis $\boldsymbol{\beta}$ gives one polynomial over E, so this term is not multiplied by k. With evaluation trace, the fold-evaluation row is already a dimension-d_A row and is covered by the common bound.

Quotient-Free Checking via Transpose Convolution The quotient method accounts for reduction modulo X^d+1 by adding a private polynomial. We now ask whether the verifier can account for that reduction directly in its public weights.

Fix a public multiplier A. Multiplication by A is linear in the witness coefficients, so it is enough to understand what happens to one basis monomial X^j. We multiply A by X^j, reduce modulo X^d+1, and evaluate the result at α. This gives the public weight assigned to the witness coefficient w_j. Adding the weighted coefficients then gives the evaluation of the reduced product, with no private quotient to carry into the next witness.

Reduction before evaluation. Fix one row, abbreviate its dimension by $d := d_r$, and write $R_d := K_r[X]/(X^d+1)$. For any $P \in K_r[X]$, the expression $P \bmod (X^d+1)$ below means the unique remainder of degree less than d; this convention makes its subsequent evaluation at α unambiguous. Given a public multiplier $A \in R_d$ and $\alpha \in E$, we define its *residue kernel* by

$$\kappa_{A,\alpha}^{(d)}(j) := \Big(A(X)X^j \bmod (X^d+1)\Big)(\alpha), \qquad j \in [d]. \tag{150}$$

Write $A(X) = \sum_{k=0}^{d-1} a_k X^k$. Since $k + j \le 2d - 2$, a monomial wraps around the modulus at most once. The kernel is therefore

$$\kappa_{A,\alpha}^{(d)}(j) = \sum_{\substack{k \in [d] \\ k+j<d}} a_k \alpha^{k+j} - \sum_{\substack{k \in [d] \\ k+j \ge d}} a_k \alpha^{k+j-d}. \tag{151}$$

The minus sign is the negacyclic wraparound $X^d = -1$. For example, with $d = 4$, $A = X^3$, and $j = 1$, the kernel is $\kappa_{X^3,\alpha}^{(4)}(1) = -1$ rather than α^4.

We have described the weight of one witness coefficient. The following identity shows that summing these contributions checks the entire 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 $\alpha \in E$,*

$$\Big(A(X)W(X) \bmod (X^d + 1)\Big)(\alpha) = \sum_{j=0}^{d-1} w_j \kappa_{A,\alpha}^{(d)}(j). \tag{152}$$

Proof. 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 (150).

One ring row. We now apply the identity to each pair of a public multiplier and a witness polynomial in a row, add their contributions, and subtract the public target. For the normal form (143), Lemma 7.3 gives the field residual

$$z_r(\alpha) := \sum_{c \in \mathsf{Occ}_r} \sum_{j=0}^{d_r - 1} \kappa_{A_{r,c},\alpha}^{(d_r)}(j) w_{\mathsf{addr}_{r,c}(j)} - Y_r(\alpha). \tag{153}$$

We reserve the uppercase Z_r for the ring residual in (143); the lowercase $z_r(\alpha)$ is its field evaluation. The quotient-free reduction checks $z_r(\alpha) = 0$ for every ring-valued row of Figure 6, at that row's native dimension. The scalar opening row remains an ordinary field-linear row.

Efficient weights and row batching. The kernel definition gives the correct weight of every witness coefficient. Computing all weights separately would take quadratic work in d. Consecutive weights instead satisfy a wraparound recurrence, which computes the complete kernel in $O(d)$ extension-field operations without division. Appendix B.1 gives this recurrence and its relation to the rotation-matrix formulation of Grand Danois [56].

We combine the resulting ring rows with the scalar opening rows using the same multilinear row batch as 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 $m_{\alpha,\tau_1}^{\mathsf{qf}}(u)$ be the sum of the batched coefficients of witness entry u, and let $V_{\alpha,\tau_1}^{\mathsf{qf}}$ be the corresponding batched public target. The complete field relation is

$$\sum_{u=0}^{2^{\mu'}-1} w_u m_{\alpha,\tau_1}^{\mathsf{qf}}(u) = V_{\alpha,\tau_1}^{\mathsf{qf}}. \tag{154}$$

Every use of a witness coordinate contributes, including uses under different native ring dimensions. The exact coefficient vector and the algorithm for evaluating its multilinear extension at the final sum-check point are given in Appendix B.1. They account for the witness addresses without adding quotient witnesses.

Completeness and soundness. The reduction has a smaller evaluation degree than quotient lifting because it tests the reduced residual itself.

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 (140). Then the following statements hold.*

1. *If every ring row of (143) and the selected scalar opening row hold, then (154) 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

$$\frac{d_{\max} - 1 + s_{\mathsf{rel}}}{|E|}, \tag{155}$$

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

Proof. 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 no false ring row is hidden by α, or if an ordinary field row is false, the resulting 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 (155). Finally, $d_{\max}$ evaluations at distinct values of α determine every degree-$< d_{\max}$ reduced residual.

No step divides by $\alpha^{d_r} + 1$. Points that are roots of one of the cyclotomic moduli therefore require no rejection and do not create an exception to completeness.

Protocol consequence. The outgoing commitment is bound before α, and the schedule digest fixes which reduction is used. The transcript then samples the existing α, τ_0, and τ_1 challenges in their prescribed order and runs the same fused sum-check as the quotient-lift route. No proof message or challenge is added. This order is essential: if the row-combination challenge were sampled before the outgoing witness was bound, the prover could choose a nonzero residual in the resulting linear kernel. The first public version of Grand Danois contained precisely this ordering issue; the current version removes the affected reduction (Appendix F.1). Every quotient interval is absent from the outgoing witness of (106). Compression digits, scalar opening rows, tensor reduction, range checks, and norm checks remain unchanged.

7.4 The Fused Sum-Check

We now bring the algebraic checks and the range check back to the same committed witness. The ring equations have been reduced to a field relation involving the witness entries. The range check has produced the claim $s_{\mathsf{claim}} = \widetilde{\mathbf{s}}(\mathbf{r}_{\mathsf{virt}})$ about the multilinear polynomial interpolating the derived Boolean values $\mathbf{s}(x) = \mathbf{w}(x)(\mathbf{w}(x)+1)$ (Section 6.1). These values were not committed separately, so we still need to bind the claim on $\widetilde{\mathbf{s}}$ to the witness we committed to.

Both obligations can be expressed as sums over the Boolean coordinates of the same witness vector. We combine them with a fresh random coefficient and run one sum-check. When compression digits are present, we also include the constraint $w(x)(w(x)+1) = 0$ at those positions, which restricts them to $\{-1, 0\}$. The resulting check connects the fold equations, the range-check computation, and the compression digits to the same witness.

The selected ring check determines one public multilinear row weight and one public target. On the quotient-lift route, let

$$m_{\tau_1}(x) := \sum_i \widetilde{\mathsf{eq}}(\tau_1, i)\, \widetilde{\mathsf{M}(\alpha)}(i, x), \qquad V_{\alpha,\tau_1} := \sum_i \widetilde{\mathsf{eq}}(\tau_1, i)[\mathbf{y}(\alpha)]_i, \tag{156}$$

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

$$m_{\tau_1}(x) := \widetilde{m^{\mathsf{qf}}_{\alpha,\tau_1}}(x), \qquad V_{\alpha,\tau_1} := V^{\mathsf{qf}}_{\alpha,\tau_1}, \tag{157}$$

using (290) and (291). Both definitions combine every row of Figure 6: commitment, fold, chain, and scalar evaluation rows alike. The verifier samples $\gamma \leftarrow E$ and, when $\mathcal{I}_{\mathsf{bin}} \neq \varnothing$, a distinct $\zeta_{\mathsf{bin}} \leftarrow E$. If $\mathcal{I}_{\mathsf{bin}} = \varnothing$, we set $\zeta_{\mathsf{bin}} = 0$ and $\widetilde{\mathsf{eq}}_{\varnothing} = 0$, so the binary-support term vanishes. Both parties run a single sum-check for

$$\begin{aligned} V_{\alpha,\tau_1} + \gamma\, s_{\mathsf{claim}} = \sum_{x\in\{0,1\}^{\mu'}} \Big[& \widetilde{\mathbf{w}}(x)\, m_{\tau_1}(x) \\ & + \Big(\gamma\, \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{virt}}, x) + \zeta_{\mathsf{bin}}\, \widetilde{\mathsf{eq}}_{\mathcal{I}_{\mathsf{bin}}}(\mathbf{r}_{\mathsf{virt}}, x)\Big) \\ & \cdot \widetilde{\mathbf{w}}(x)\big(\widetilde{\mathbf{w}}(x)+1\big)\Big], \end{aligned} \tag{158}$$

which simultaneously (i) verifies every relation row (the evaluation claim included) and (ii) binds the claim on $\widetilde{\mathbf{s}}$ to the committed witness. The additional binary constraint applies only to the compression digit intervals; its restricted equality weights are evaluated as described in Remark 6.2. The relation term $\widetilde{\mathbf{w}}(x)m_{\tau_1}(x)$ does not carry the range term's equality factor. The complete round message therefore uses the standard compressed format rather than the equality-factored format. Let $\mathbf{r}_2$ be the final sum-check point and v' the claimed witness value there. Together with the public weights, this value determines the bracketed summand of (158) at $\mathbf{r}_2$. After this check, the remaining private claim is the opening $\widetilde{\mathbf{w}}(\mathbf{r}_2) = v'$ passed to the next level.

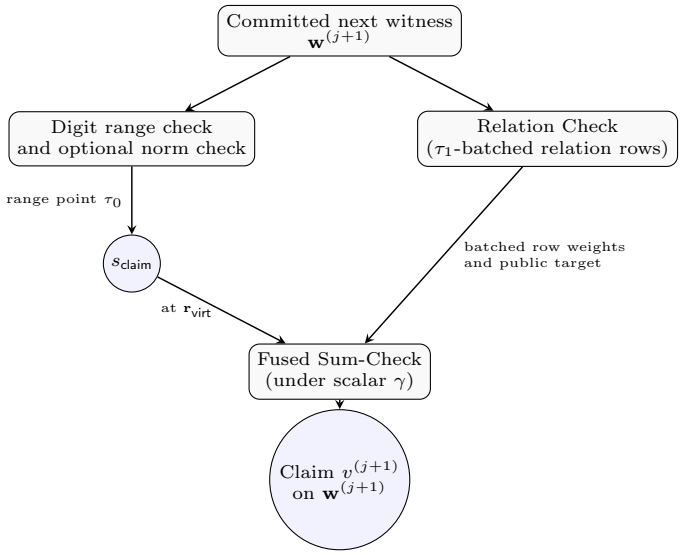

Fig. 7: The checks at one fold. The digit-range check (Section 6.1) supplies a claim that joins the relation check under γ (Section 7.4). An admissible Euclidean route also certifies the complete reconstructed response norm through these checks (Section 6.2).

7.5 Verifier Evaluation of the Batched Row

At the final sum-check point $\mathbf{r}_x = \mathbf{r}_2$, we must evaluate $m_{\tau_1}(\mathbf{r}_x)$ without constructing its $2^{\mu'} = \Theta(|\mathbf{w}|)$-column relation matrix. On the quotient-lift route this is the multilinear extension of the weighted rows of $\mathsf{M}(\alpha)$; on the quotient-free route it is the sum of the native row occurrences in (294) and the scalar rows. We precompute the row weights $\widetilde{\mathsf{eq}}(\tau_1, i)$ and separate structured public factors from coefficients of the shared setup.

Table 6: 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.

Structured weights and shared setup. The non-matrix blocks combine public factors such as folding challenges, gadget powers, and equality weights. We evaluate these factors at the witness addresses using Appendix A. Offsets and incomplete intervals can introduce carries between coordinates; the evaluator accounts for those carries without enumerating one public coefficient per witness coordinate. Appendix B.3 specifies the factors for each opening method and ring-reduction route.

The matrix blocks depend on the shared setup coefficients themselves. Let $\mathcal{M}_{\mathsf{setup}}$ index the active $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{D}$ matrix views, and define their longest prefix by $N_{\mathsf{active}} = \max_{I \in \mathcal{M}_{\mathsf{setup}}} n_I m_I d_I$. If coefficient S_p occurs in several views, we add all its weights before reading it. Writing this combined weight as $\bar{\omega}(p)$, their total contribution is

$$\sigma_S = \sum_{p < N_{\mathsf{active}}} S_p \bar{\omega}(p). \tag{159}$$

Only $\mathbf{B}$ has multiple logical slices. Those slices can reuse the same matrix coefficient, so all their occurrences enter its combined weight. Constructing that weight also costs work: the structured evaluator contracts the slice and address factors rather than explicitly expanding every use. Appendix B.4 gives the exact weights and their costs.

Cost. The factored-block cost is the equality-weight preparation and carry-state contraction cost of the native affine maps (Appendices A.4 and B.3), plus the outer combination steps. The fused scan performs N_{active} base-field-by-extension-field multiply-adds, plus the transition work needed to evaluate the matrix weights and the $\mathbf{B}$-slice contraction. Quotient-free checking additionally prepares the terminal recurrence for each distinct native address geometry. If D_{term} is the sum of the dimensions of these geometries, this preparation costs $O(D_{\mathsf{term}})$ extension-field operations and storage; it does not depend on the witness length. Admissibility bounds both the number of relation-row groups and this transition work. Under those bounds, the setup scan retains the square-root profile when the active setup width does. There is also the direct compression-matrix 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, in addition to contracting its sparse relation weights. This work remains local even at an offloaded level. On the quotient-lift prefix, verifier offloading (Section 9) may replace the $\mathbf{A}/\mathbf{B}/\mathbf{D}$ scan by the setup-product sum-check. The current admissibility rules require the quotient-free suffix to use the direct scan; this is a schedule restriction, not an algebraic requirement of Lemma 7.3.

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 input record. 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 j, the parties hold $\widetilde{\mathbf{w}}^{(j+1)}(\mathbf{r}^{(j+1)}) = v^{(j+1)}$ under the commitment absorbed during that fold. Any offloaded edge also carries $\mathcal{S}_j$ according to (54). The final fold instead binds the terminal inner state defined below. We still need to specify the opening method at each receiver.

The *analyzed opening-mode policy* derives the opening method from the absolute level j and transition kind:

$$\mathsf{method}(j, \mathsf{kind}) := \begin{cases} \mathsf{SubringCoefficientPacking} & \mathsf{kind} = \mathsf{nonterminal},\ j \in \{0,1\}, \\ \mathsf{EvaluationTrace} & \text{otherwise.} \end{cases} \tag{160}$$

This rule does not require levels 0 or 1 to exist. It assigns a method only to the folds already present in the schedule.

Both opening methods leave an ordinary base-field witness opening. The receiving fold performs its opening-method reduction, so switching to evaluation trace requires no extra bridge message from the preceding fold. A final packing fold can therefore lead directly to the evaluation-trace terminal.

8.2 Terminal Levels (Tail Handling)

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. The terminal takes this same state as public input. This specialization is what permits the terminal to omit the outer $\mathbf{B}$ relation; simply dropping that relation while retaining an outer commitment as the public state would be unsound.

The terminal accepts one recursive-witness group and no pending setup opening. The final fold must therefore check any incoming setup group and produce none. It may use either opening method: even a root-only packing schedule can be followed by this evaluation-trace terminal.

Reducing the incoming opening. Having fixed the terminal's public input, we next describe the values used by its direct checks. The tensor reduction turns the incoming opening into one packed claim; the terminal then forms its partial opening evaluations and folded response. Let

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

The terminal's single tensor-reduction prefix produces a packed claim $g_{\mathsf{term}}(\rho_{\mathsf{term}}) = \bar{v}_{\mathsf{term}}$ 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 requires $\theta_{\mathsf{term}} \neq 0$ and rejects without resampling if the factor vanishes. This event is separate from response grinding because it occurs before the terminal response nonce. Split the tensor-reduction output point as

$$\rho_{\mathsf{term}} = (\rho_{\mathsf{blk,term}}, \rho_{\mathsf{pos,term}}, \rho_{\mathsf{pack,term}})$$

over the block, within-block position, and trace-packing axes. Define the terminal position vector and block weights by

$$a_{\mathsf{term},x} := \widetilde{\mathsf{eq}}(\rho_{\mathsf{pos,term}}, x), \qquad \chi_{\mathsf{blk,term}}(i) := \widetilde{\mathsf{eq}}(\rho_{\mathsf{blk,term}}, i).$$

The trace functional $\mathcal{T}_{\rho_{\mathsf{pack,term}}}$ is the specialization of (130) to the terminal packing point. Set

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

These are the terminal specializations of (93) and the evaluation-trace folded response.

Revealed values and direct checks. The message order matters because 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 clear partial opening evaluations $(e_i)_i$, and then runs the scheduled response grind. Each candidate nonce is absorbed before the transcript derives the complete fold-challenge tuple $(c_i)_i$. A nonce is admissible only when every bounded fixed-filter 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

$$\mathbf{A}_{\mathsf{term}} \mathbf{z} = \sum_{i \in [B_{\mathsf{term}}]} c_i \mathbf{t}_i \qquad \text{in } R_{\mathsf{term}}^{n_{A,\mathsf{term}}}, \tag{161}$$

$$\sum_{i \in [B_{\mathsf{term}}]} c_i(X) e_i(X) = \mathbf{a}_{\mathsf{term}}(X)^{\top} \mathbf{G}_{b_{\mathsf{term}}, M_{\mathsf{term}}} \mathbf{z}(X) \qquad \text{in } R_{\mathsf{term}}, \tag{162}$$

$$\theta_{\mathsf{term}} \sum_{i \in [B_{\mathsf{term}}]} \chi_{\mathsf{blk,term}}(i)\, \mathcal{T}_{\rho_{\mathsf{pack,term}}}(e_i) = v_{\mathsf{tensor}} \qquad \text{in } E. \tag{163}$$

These specialize the fold-evaluation and trace rows (136) and (131). 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 (161) and (162) 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

$$\begin{aligned} \mathbf{t}_{B-1} &= c_{B-1}^{-1} \Big(\mathbf{A}_{\mathsf{term}} \mathbf{z} - \sum_{i < B-1} c_i \mathbf{t}_i \Big), \\ e_{B-1} &= c_{B-1}^{-1} \Big(\mathbf{a}_{\mathsf{term}}^{\top} \mathbf{G}_{b_{\mathsf{term}}, M_{\mathsf{term}}} \mathbf{z} - \sum_{i < B-1} c_i e_i \Big). \end{aligned} \tag{164}$$

The verifier can therefore compute the last block instead of receiving it. Computing it alone would prove nothing, since (164) makes the two checks hold by construction; the omitted values must still be fixed before the challenges that test them. We fix them with a digest. 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. The transcript changes in two places.

- The predecessor sends $(\mathbf{t}_i)_{i<B-1}$ and $\delta_{\mathbf{t}} = \mathcal{H}_{\mathsf{dig}}(\mathbf{t}_{B-1})$ as its outgoing binding, at the position where it sent the full state.
- The terminal sends $(e_i)_{i<B-1}$ and $\delta_e = \mathcal{H}_{\mathsf{dig}}(e_{B-1})$, at the position where it sent the full partials: after its tensor reduction and before the response nonce and $(c_i)_i$.

After receiving $\mathbf{z}$, the verifier computes the right-hand sides of (164), rejects unless they hash to $\delta_{\mathbf{t}}$ and δ_e, and runs the trace check (163) over all B partials, including the recomputed one. The verifier multiplies each recomputed value back by c_{B-1} and checks (161) and (162) 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 10.45 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 and encoding. 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. 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.[12] The verifier rejects a nonce outside the scheduled set, a failed fixed-filter sample, and a response outside these bounds. 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 [64].

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 contribution σ_S in (159). We replace that scan by a sum-check whose output is an opening of a precommitted setup prefix. The next fold checks this opening beside its recursive-witness claim, using the same batch fold as before. The smaller $\mathbf{F}/\mathbf{H}$ maps remain local; we offload only the $\mathbf{A}/\mathbf{B}/\mathbf{D}$ contribution.

9.1 The Setup Claim and Setup-Product Sum-Check

To turn the setup work into an opening claim, we first isolate the matrix-dependent part of the verifier's final row evaluation. 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 row evaluation separates as

$$m_{\tau_1}(\mathbf{r}_2) = m_{\mathsf{local}}(\mathbf{r}_2) + \sigma_S. \tag{165}$$

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 the contributions read from the shared setup vector through the $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{D}$ views only.

We use the active setup-prefix length N_{active} and the combined weight $\bar{\omega}$ from Section 7.5. The setup source is represented on a padded Boolean domain of length 2^{ν_S} with $2^{\nu_S} \ge N_{\mathsf{active}}$. Write

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

for this padded source. The row-batching point τ_1, the relation point $\mathbf{r}_x$, the 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 instead uses the transposed residue kernel of Section 7.3. Contributions from all overlapping views are added inside the same weight, as in (159), and $\omega_{\mathsf{setup}}(p) := 0$ for $N_{\mathsf{active}} \le p < 2^{\nu_S}$. The delegated value is therefore

$$\sigma_S = \sum_{p < 2^{\nu_S}} \mathbf{S}^\flat(p)\, \omega_{\mathsf{setup}}(p). \tag{166}$$

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

[12] An implementation may instead impose a tighter encodability limit; response admission then also checks that the encoding fits this limit.

Without offloading, the verifier evaluates (166) by scanning the active prefix. With offloading, the prover supplies a claimed σ_S for the final check of Stage 2. The first two stages are otherwise unchanged: Stage 1 performs the digit-range check of Section 6.1, and Stage 2 performs the fused relation and range sum-check of Section 7.4. Stage 2 also produces the next-witness opening

$$v_W := \widetilde{\mathbf{w}^{(h+1)}}(\mathbf{r}_2).$$

Once Stage 2 has fixed $\mathbf{r}_x$, the setup weight in (166) is fully determined. Stage 3 then proves the claimed setup inner product by running a degree-2 sum-check for

$$\sigma_S = \sum_{p \in \{0,1\}^{\nu_S}} \widetilde{\mathbf{S}^{\flat}}(p)\, \widetilde{\omega}_{\mathsf{setup}}(p), \tag{167}$$

where ν_S is the logarithm of the padded setup-domain length. The sum-check samples a point ρ_S and ends with

$$(\text{sum-check output}) = s_{\rho_S}\, \widetilde{\omega}_{\mathsf{setup}}(\rho_S), \qquad s_{\rho_S} := \widetilde{\mathbf{S}^{\flat}}(\rho_S). \tag{168}$$

The round messages use the standard compressed format, as recorded in Table 6. The verifier evaluates the public weight without scanning the setup prefix. The remaining value s_{ρ_S} is carried to the next fold.

9.2 Passing the Setup Opening to the Next Fold

Stage 3 leaves an opening claim rather than checking the setup polynomial at ρ_S: evaluating it directly would restore the scan we meant to avoid. We pass that claim to the successor in the scheduled order

$$\mathcal{O}_{h+1} = (\mathcal{S}_h, \mathcal{W}_{h+1}), \qquad \mathcal{S}_h = (C_{S,h}, \rho_S, s_{\rho_S}), \quad \mathcal{W}_{h+1} = (C_{W,h+1}, \mathbf{r}_2, v_W). \tag{169}$$

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

$$v_{\mathsf{boundary}} := \phi_S s_{\rho_S} + \vartheta_{\mathsf{grp}} \phi_W v_W, \tag{170}$$

where each padding factor uses its own group's domain. With evaluation trace, the shared tensor reduction first produces the scaled claims $h_S = \theta_S \bar{s}_{\rho_S}$ and $h_W = \theta_W \bar{v}_W$ from the padded inputs, where $\bar{s}_{\rho_S}$ and $\bar{v}_W$ are the packed output values. Their checked nonzero factors include the native-prefix adjustments of Section 5.1, and the target is

$$\bar{v}_{\mathsf{boundary}} := h_S + \vartheta_{\mathsf{grp}} h_W. \tag{171}$$

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. Thus at most one setup opening is pending at any time, and the last nonterminal fold must check its incoming opening and produce none. The terminal checks its remaining matrix work locally on its one recursive-witness state (Section 8.2).

Lemma 9.1 (Soundness of setup offloading). *Fix an offloaded level with active setup length N_{active}. Conditioned on the soundness of the Stage 3 product sum-check and on binding of the successor's accepted opening of the registered commitment $C_{S,h}$, the value σ_S used in Stage 2 equals the verifier's direct fused setup scan (159). The binding condition is established by the source-comparison class $C^{\mathsf{cmp}}_{S,h}$ containing the setup occurrence. Its $\mathbf{A}$ instance has radius $\eta_{A,C^{\mathsf{cmp}}_{S,h}}$ from (197), and its $\mathbf{B}/\mathbf{F}$ instances use (198). This class contains both the successor's protocol occurrence and the canonical reference occurrence $u^0_{S,h}$ of (194). Thus its radius covers the mixed comparison between the extracted opening and the setup's canonical algebraic decommitment, rather than only two extractions under the successor profile.*

Proof. 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 (193). 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 $\widetilde{\mathbf{S}^\flat}(\rho_S)$. Soundness of Stage 3 then fixes σ_S to the inner product of that source and ω_{setup}. The weight is zero outside the active range and agrees with $\bar{\omega}$ of (159) inside it, so the two sums are identical.

Evaluating the Public Setup Weight Stage 3 leaves the public factor $\widetilde{\omega}_{\mathsf{setup}}(\rho_S)$ in (168). The verifier computes it 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 $\mathbf{B}$; $\mathbf{A}$ and $\mathbf{D}$ remain unsliced. Appendix B.4 gives the contraction using the general evaluator of Appendix A.

Size of the Carried Setup Group The setup prefix is a second opening group at the next fold. Under the rank-one profile, its active field-coefficient length is controlled by the recursive witness. If m_I and d_I are the width and ring dimension of $I \in \mathcal{M}_{\mathsf{setup}}$, then the source digits of each rank-one view occupy $d_I m_I$ coordinates in the successor. Write $|\mathbf{w}'|_K$ for the number of base-field coordinates in that successor witness. Since $n_I = 1$ in this profile, the definition of N_{active} in Section 7.5 specializes to

$$N_{\mathsf{active}} = \max_{I \in \mathcal{M}_{\mathsf{setup}}} d_I m_I \le |\mathbf{w}'|_K. \tag{172}$$

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

$$N_{\mathsf{prefix}} < 2N_{\mathsf{active}} \le 2|\mathbf{w}'|_K. \tag{173}$$

For larger module ranks, schedule validation checks the exact rounded shape instead of relying on this inequality. The planner uses the same exact shape when deciding which nonterminal levels offload; no fixed offloading depth is part of the protocol.

9.3 Full Schedules and Admissibility

A schedule must connect valid folds and supply the bounds used by the security proof. This is separate from choosing which combinations of opening methods, compression, and ring checks to implement. We state the recursive interface and the family analyzed in this paper first, then give the compatibility and security conditions that define admissibility.

Recursive interface. The root source $\mathcal{O}_0$ is a nonempty list of application-supplied opening groups satisfying the root batching conditions of Section 5. No application-supplied group occurs at a later level. After a direct edge, the next source is exactly $(\mathcal{W}_{j+1})$; after an offloaded edge, it is exactly $(\mathcal{S}_j, \mathcal{W}_{j+1})$. The final edge is TerminalInnerState and supplies the one recursive-witness state required by Section 8.2.

Schedule family analyzed here. The following restrictions simplify the implemented schedules and their analysis; they are not intrinsic requirements of the commitment or offloading identities. We retain them as explicit scope for the results that assume an admissible schedule. In particular, the restriction on Euclidean norm proofs specifies which response configurations our current analysis covers.

(i) *Opening and norm methods.* Nonterminal folds at levels 0 and 1 use direct coefficient packing and coefficient-ℓ_∞ response bounds; later folds use evaluation trace with tensor reduction. This rule applies only to folds that exist in the schedule. The Euclidean route applies to one recursive-witness response at an evaluation-trace stage, after setup openings and response chunking have ended, or at the terminal.

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

(iii) *Ring checks.* Quotient lifting forms 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. Their outgoing claims therefore go directly to the next witness fold or terminal. The terminal checks its clear response directly and has no ring-check tag.

In particular, excluding quotient-free offloading is a simplifying choice. The setup-product identity (166) also applies to quotient-free rows, using the transposed residue kernel in place of quotient-lift weights. The public-weight evaluation cost changes, but the reduction to an opening of the setup prefix is the same.

Compatibility between folds. Direct packing requires $d_A = khd_{\mathsf{car}}$ and the same challenge in the subring consistency equation and the embedded $\mathbf{A}$ equation. A Euclidean certificate must bound the complete reconstructed response used by that equation. These requirements concern the local relations, rather than the level at which a method is chosen.

Every level's fold description must satisfy the matrix-dimension, coefficient-alphabet, and challenge conditions of Sections 4.1, 4.3, 4.5 and 5. For a multi-group source, the corresponding conditions apply to every group, and the shared $\mathbf{D}/\mathbf{H}$ layout must be the one derived in Section 5. Every $\mathbf{B}$ slice count satisfies the commitment-slicing conditions of Section 4.5; every target shape is exactly the source shape used at the next level; and every committed segment is written in an alphabet certified by the level that scans it. In particular, each group satisfies $b_{1,g} \le b^\star$ for opening and inner-image digits (Remark 10.24) and $b_{z,g} \le b^\star$ for response digits (Section 5.5), where $b^\star$ is the certified range base of the scanning level. Quotient and compression digits satisfy their prescribed alphabet conditions as well. For an offloaded edge, the producer's Stage 3 statement, the commitment and frozen source profile in the schedule-selected registry slot, and the successor's setup group must agree exactly. The setup group is a source occurrence at that successor. Its frozen $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{F}$ ranks must meet the security floors for the comparison class containing that occurrence, under the receiving fold's certified challenge, response, and digit profiles. 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 evaluation-trace claim coefficients only after the shared opening payload is fixed, with the first coefficient normalized to one. A fold may check one setup group and produce a new one, but the two groups do not accumulate. The final committed fold must check any incoming setup group and cannot produce another one for the terminal.

Security conditions. Every Module-SIS instance induced by these maps and their certified collision bounds must meet the stated security floor, with response-derived norms taken from (110). These instances use the schedule-defined source-comparison classes of (197), including the corresponding cross-profile source-path alphabet diameters.

Every accepted challenge family must have a certified cardinality lower bound and certified pairwise-difference conditions. The protocol derives each multi-coordinate folding tuple by bounded fixed-filter sampling. By Lemma 10.1, conditioned on sampler success, its random-oracle distribution is the product of the corresponding accepted families. For every challenge round r and coordinate u, the schedule records $L_{r,u}$, a certified lower bound $\underline{\alpha}_{r,u}$, and

$$\underline{a}_{r,u} := 1 - (1 - \underline{\alpha}_{r,u})^{L_{r,u}}. \tag{174}$$

We require $L_{r,u}$ and $1/\underline{a}_{r,u}$ to be polynomially bounded. The transcript program's worst-case honest $\mathcal{H}_{\mathsf{FS}}$ query count, including all allowed nonce probes, must not exceed $Q_{\max}$. The schedule must satisfy

$$\mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch}) \le T_{\mathsf{poly}}(\lambda, N_{\mathsf{exp,max}}), \tag{175}$$

where $\mathsf{Tree}_{\mathsf{ck}}$ is the tree size of the checkpoint extractor, defined in (228), which is at most the size of the complete mixed tree in (227), and $N_{\mathsf{exp,max}}$ is the environment's maximum total explicit length of the public instance and extracted witness, measured in base-field coordinates, over an admitted schedule. In particular,

$$N_{\mathsf{exp,max}} \ge 2^{\mu_{\max}}.$$

The bound T_{poly} is a fixed polynomial in the security parameter and this explicit length. It need not be polynomial in the logarithmic variable count $\mu_{\max}$, since an extractor must already spend linear time to output a dense witness. Finally, the exact schedule-wide interactive error must satisfy the Fiat–Shamir budget (226) for the recorded $(\lambda_{\mathsf{FS}}, Q_{\max})$. Selecting a raw challenge family does not suffice unless this full check passes.

Definition 9.2 (Admissible Akita schedule). *A schedule in the family specified above is* admissible *if:*

(i) *its folds and terminal follow the stated recursive interface;*

(ii) *its commitments, digit alphabets, and outgoing claims satisfy the compatibility conditions above; and*
(iii) *its matrix instances, challenge samplers, extraction tree, and transcript error satisfy the security conditions above.*

Implementing the challenge sampler. For a nonce, round, and coordinate, the ideal coordinate oracle supplies a lazy uniform bit stream. Here $L_{r,u}$ counts completed shell draws, not bit words: exact bounded-integer rejection within a draw consumes a variable number of bits. The implementation uses a coordinate-indexed SHAKE stream from the group's transcript seed. We analyze that expansion in the ideal-XOF model, with the coordinate stream as the programmable object. Hence Lemma 10.2 gives the exact conditional preimage sampler required by the classical-ROM tree builder. An implementation that obtains raw draws from separate $\mathcal{H}_{\mathsf{FS}}$ counter queries must count every such query instead. Sampler failure rejects that nonce attempt.

Admissibility is a condition on the resulting sequence, not on how it was found. Soundness neither assumes that a planner found an optimum nor depends on its cost model.

9.4 The Constant-Root Verifier

To obtain a constant-root verifier, the first few folds use a different geometry from a balanced direct schedule. Fix a constant $\kappa_{\mathsf{root}} \geq 2$ and set the target scale

$$N_\star := \left\lceil N^{1/\kappa_{\mathsf{root}}} \right\rceil .$$

For the first $\kappa_{\mathsf{root}} - 1$ folds, choose $B_j = \widetilde{\Theta}_{\kappa_{\mathsf{root}},\lambda}(N_\star)$ source blocks and block length $M_j = \lceil N_j/B_j \rceil$, up to the required power-of-two rounding. The verifier processes the B_j challenge-dependent contributions and directly evaluates the $\mathbf{F}/\mathbf{H}$ matrices, while setup offloading removes the scan over the longer block. The regularity conditions below bound the remaining local work. This asymptotic prefix need not coincide with the at-most-two-fold packing prefix. When $\kappa_{\mathsf{root}} > 3$, some of these wide folds use evaluation trace. The argument below uses only the closed successor size and the regularity conditions, so it applies under either opening method.

Admissibility alone is a soundness condition and does not imply the size recurrence below. We therefore call a family of admitted schedules *constant-root regular* if, at every one of these early levels:

1. 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 $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(1)$;
2. all 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;
3. every successor segment other than the folded response has total length $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(B_j)$;
4. the longest active flat $\mathbf{A}/\mathbf{B}/\mathbf{D}$ setup prefix satisfies the exact rounded balance check, with

$$N_{\mathsf{prefix},j} = \widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(|\mathbf{w}_{j+1}|);$$

5. 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. In particular, any factor over the M_j positions is evaluated through its equality structure, without enumerating all M_j values.

These conditions concern a family as N grows and are checked on its exact successor layouts. A fixed matrix and compression template is sufficient only if it remains admissible throughout that family and its derived quantities satisfy the displayed growth bounds. A schedule validated at one input size does not establish this uniform statement. In particular, any growth needed to maintain the challenge-entropy, field-error, and Module-SIS budgets must be included in the regularity check. The conditions exclude schedules whose slice count, chain depth, norm-claim count, or matrix rank grows polynomially with N.

The direct compression work is already controlled by these parameters. The first $\mathbf{F}$ map acts on digits of the stacked $\mathbf{B}$ image, which has $S_B n_B d_B$ coefficients, and the first $\mathbf{H}$ map acts on digits of the $\mathbf{D}$ image, which has $n_D d_D$ coefficients (Section 4.5). Each later map acts on digits of the preceding compression image. Thus the bounds on slice counts, module ranks, ring dimensions, digit depths, and chain lengths already make the total $\mathbf{F}/\mathbf{H}$ evaluation work and expanded storage $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(1)$. Leaving these maps with the verifier requires no additional regularity condition.

The folded response has M_j positions, and the remaining data attached to the source blocks has size $\widetilde{O}_\lambda(B_j)$. The carried setup group is bounded as in Section 9.2. Hence the witness satisfies

$$N_{j+1} = \widetilde{O}_{\kappa_{\mathsf{root}},\lambda}\left(\frac{N_j}{N_\star} + N_\star\right). \tag{176}$$

Induction gives $N_j = \widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N/N_\star^j + N_\star)$, so after $\kappa_{\mathsf{root}} - 1$ folds the witness has size $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N_\star)$. The core recursion then continues to the terminal check of Section 8.2.

At every early level, the verifier performs $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N_\star)$ work. Once the witness reaches this size, a direct check of the remaining data has the same bound. The early folds add a constant number of logarithmic transcripts, and the witness sizes at successive levels form a geometric series beginning at N.

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 whose first $\kappa_{\mathsf{root}} - 1$ folds use the geometry above and setup offloading, followed by balanced direct folds retaining the polylogarithmic parameter and per-level count bounds above. Require $N_{j+1} \leq N_j^\beta$ 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_j includes any incoming setup group. Use the coefficient-ℓ_∞ response route in this asymptotic family. More generally, the statement also holds if 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, 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, including $\mathbf{A}$, $\mathbf{B}$, $\mathbf{D}$, and any $\mathbf{F}/\mathbf{H}$ maps, in addition to the setup seed and the short commitments authenticating the offloaded prefixes.*

Proof. The recurrence (176) reduces the witness to $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N_\star)$ after $\kappa_{\mathsf{root}} - 1$ folds. At each early level, the challenge scan costs $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N_\star)$ and setup offloading removes the only scan whose width can be M_j. The direct suffix begins at total source size $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N_\star)$. Its parameter bounds and contraction give $\sum_j N_j = \widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N_\star)$ and $\sum_j \log N_j = O(\log N)$ over the suffix. The first sum bounds its verifier work and expanded setup storage; the second bounds its transcripts under the stated response-route condition. The terminal contributes only polylogarithmic cost. A digit-expanded norm check with an unbounded number of response segments would add those claims to the proof and is therefore a concrete optimization, not part of this asymptotic family. Summing the total source lengths, including carried setup groups, over the constant early prefix and this suffix gives the prover bound. Verifiers at the first $\kappa_{\mathsf{root}} - 1$ producer levels do not require expanded $\mathbf{A}/\mathbf{B}/\mathbf{D}$ prefixes, since these contributions are offloaded. They still evaluate the $\mathbf{F}/\mathbf{H}$ matrices directly. As shown above, their evaluation work and expanded storage are $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(1)$. The first direct producer occurs after the witness has size $\widetilde{O}_{\kappa_{\mathsf{root}},\lambda}(N_\star)$, so its active matrix prefix, and every later one, has the same bound.

A larger fixed κ_{root} lowers the verifier exponent but adds early offloaded folds. For every fixed κ_{root}, proof size remains polylogarithmic and prover work remains linear up to polylogarithmic factors. Completeness and knowledge soundness for these schedules are covered by Section 10.

A Parameter Choice Attaining the Bound The preceding theorem separates the cost argument from parameter selection. We now give one choice of schedules that meets its conditions. The relevant hardness requirement is quantitative: as the input grows, we must retain security against algorithms that can already read that input. A hardness claim for a fixed modulus, dimension, or collision bound would not suffice.

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 parameter-growth assumption for ordinary Module-SIS, specialized to 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 the coefficient-wise Module-SIS instances of degree d, rank one, width at most 2^{CL}, and collision bound at most 2^{CL} have solving probability at most 2^{-CL} for algorithms running in time at most 2^{CL}. Enlarging C covers

the polynomial running-time overhead of Theorem 10.44 and the allocation of error among the active maps. This is an assumption about the full SIS parameter tuple, not merely an exponential attack-cost estimate in the dimension. It introduces no additional structure in 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.*

Proof. Use one root opening, a common ring degree d for all maps, module rank one, and one $\mathbf{B}$ slice. Take $E = \mathbb{F}_q$, so that the extension degree is one, and use the entire ring as the challenge subring at the first two levels. Later levels use the corresponding evaluation-trace encoding. For this parameter choice, use quotient lifting throughout the offloaded prefix and while checking its final setup opening. Its factored weights simplify the cost analysis; this choice is sufficient for existence and does not assert that offloading requires quotient lifting. A later direct suffix may use quotient-free checks. Fix the balanced digit base $b = b_1 = 4$, with no Euclidean norm certificates, operator filtering, or response-dependent grinding. Root and setup-prefix commitments use the prescribed two binary stages; their outputs need not have the concrete 128-byte size. Keep compression throughout the offloaded prefix and its last receiver, then use raw commitments for the direct suffix. This makes only one compressed-to-raw transition. Every ring and digit conversion has $L^{O(1)}$ overhead. Full-field source coefficients require $O(\log q)$ base-four digits. Quotient coefficients have bit length $O(\log m + \log q + \log d)$, so their digit depths are also polynomial in L. Setup sampling uses enough bits per coefficient to make the total sampling bias negligible under Section 4.3.

For challenges, use all sign vectors with exactly $d/2$ nonzero entries. Their number is $\binom{d}{d/2}2^{d/2} = 2^{\Omega(d)}$, 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; the invertibility condition of Section 3.2 holds. Increasing the constant in $d \geq CL$ covers the sum of the folding errors and the factor $Q_{\max} + 1$. The modulus also covers all field-challenge errors. For a folded digit vector with B_j blocks, the deterministic coefficient bound is $O(B_j d)$; its base-four decomposition therefore has $O(\log B_j + \log d)$ planes. In particular, neither honest completeness nor the challenge count requires a response model or an operator filter. To sample the shell exactly, draw a uniform integer index into the shell by binary rejection and unrank its support and signs. Each trial succeeds with probability at least $1/2$; allowing CL trials per coordinate gives a fixed raw-draw tape with failure probability at most 2^{-CL}. A union bound over the coordinates covers sampler failure within the completeness budget, and the accepted distribution is uniform. Unranking takes time polynomial in d. All commitment collision bounds and matrix widths are $2^{O(L)}$, so the assumed SIS bounds apply after choosing C sufficiently large.

Let $N_\star = \lceil N^{1/\kappa_{\mathsf{root}}} \rceil$. At each early fold choose a power-of-two block length within a factor two of the current group length divided by $N_\star$, with a minimum of one ring position; use separate such lengths for the at most two source groups. There are $O(N_\star)$ blocks per group. All response planes together have size $L^{O(1)} N_j / N_\star$. Inner images, quotient polynomials, compression digits, and other auxiliary digits have total size $L^{O(1)} N_\star$: ranks, ring degrees, and numbers of claims are polynomial in L, while there are only constantly many source groups. Write W_{j+1} for this ordinary successor length. The rank-one balance of (173) gives $N_{\mathsf{prefix},j} \leq 2W_{j+1}$, including power-of-two padding. The input size at the next level counts both source groups, hence $N_{j+1} = W_{j+1} + N_{\mathsf{prefix},j} \leq 3W_{j+1}$. At the next fold, apply the same block-length choice separately to these two groups. Thus

$$N_{j+1} \leq L^{O(1)}(N_j/N_\star + N_\star).$$

After $\kappa_{\mathsf{root}} - 1$ early folds the remaining size is $L^{O_{\kappa_{\mathsf{root}}}(1)} N_\star$. Use balanced direct folds thereafter, stopping when the witness has size $L^{O(1)}$ and performing the terminal check. For inputs already below this threshold, use a direct schedule from the start. Choosing the threshold large enough absorbs the digit and ring overhead in each direct fold.

The common ring dimension and consecutive digit layout leave polynomially many (in L) coordinate ranges and relation families. Each family uses only a constant number of simultaneous affine addresses. After peeling the power-of-two position strides, the remaining strides and carry ranges are polynomial in L: they involve only ring lanes and digit indices. 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 A and B.4 therefore evaluate the remaining weights in $L^{O(1)}N_\star$ operations, without scanning a long position interval. All active two-stage binary compression maps act on only $L^{O(1)}$ coefficients: each rank-one image has d coefficients and its binary expansion has $d\lceil \log_2 q \rceil$ digits. Their work satisfies the same bound. These choices verify all five regularity conditions above.

Finally, fixed digit bases give constant sum-check degree. Above a sufficiently large polynomial-in-L stopping threshold, each balanced direct fold reduces N_j to at most $N_j^{3/4}$. Consequently $\sum_j \log N_j = O(\log N)$ over this suffix. Its number of folds is $O(1 + \log(\log N/\log L))$, so their accumulated $O(\log L)$ costs are also $O(\log N)$. Each fold has $O(\log N_j + \log L)$ constant-degree rounds and polynomially many claim-combination branches in N_j and L. The fixed early prefix and the terminal contribute 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 (228), has size $(NL)^{O_{\kappa_{\mathsf{root}}}(1)}$, as required for admissibility. The exact security allocation is made using Equations (175) and (226).

Sufficient hardness assumptions. A suitable exponential hardness guarantee is sufficient but is not necessary for polynomial parameter growth. In this discussion, a hardness guarantee includes the success-probability bounds of the assumption above, rather than just the cost of one known attack. To make the distinction precise, suppose the relevant SIS instances provide $h(D)$ bits of hardness at an effective size D, and implementing this size costs a polynomial in D and L. Here h must already account for the modulus, width, collision norm, and reduction losses. We need $h(D) = \Omega(L)$. If $h(D) = \Omega(D^\alpha)$ for any fixed $\alpha > 0$, choosing $D = L^{O(1/\alpha)}$ preserves every displayed soft-O bound. This includes attack-time lower bounds of the form $2^{\Omega(\sqrt{D})}$, not just $2^{\Omega(D)}$.

If instead the available guarantee is only $2^{\Omega((\log D)^c)}$ for fixed $c > 1$, the analogous sufficient choice is $D = 2^{O(L^{1/c})}$. Polynomial overhead in this size is not polylogarithmic in N. For fixed λ and polynomial $Q_{\max}$, the same accounting gives prover cost $N^{1+o(1)}$, verifier cost $N^{1/\kappa_{\mathsf{root}}+o(1)}$, and proof size $N^{o(1)}$, rather than the polylogarithmic proof guarantee. Thus "subexponential hardness" alone is too imprecise: a fixed positive power in the hardness exponent suffices, whereas a merely quasipolynomial guarantee does not by itself justify the stronger bounds.

9.5 The Assembled Protocol

Figure 8 specifies how the setup, commitment algorithm, and canonical fold implement the PCS interface. The security analysis of Section 10 is stated against this figure.

The Akita protocol, assembled.

$\mathsf{Setup}(1^\lambda, \mu_{\max})$*:* fix an admissible schedule (Section 9.3); expand the seed into the shared setup vector $\mathbf{S}$ per (60) with its stacked prefix views (Figure 3); prepare the setup data required by every scheduled direct edge; and populate the verifier's setup-prefix registry with the exact commitment $C_{S,h}$ selected for every scheduled offloaded edge.

$\mathsf{Commit}(\mathsf{pp}, \mathsf{shape}_{\mathsf{com}}, (f_p)_{p\in[P]})$*:* run the commitment pipeline (Figure 4) with the ordered vectors prescribed by the declared root-level source layout ($P = 1$ for the base PCS interface) and output the commitment together with its shape $\mathsf{cm} = (\mathsf{shape}_{\mathsf{com}}, \mathbf{u}_{\mathsf{pub}})$. Commitments that later enter one root batch remain independently formed and retain their own shapes.

$\mathsf{Eval}(\mathsf{pp}, \mathcal{O}_0)$*:* absorb the seed identity, schedule digest including $(\lambda_{\mathsf{FS}}, Q_{\max})$, and every root commitment, point, and claimed value; then:

1. **Root and recursive folds.** Starting from $\mathcal{O}_0$, run Figure 5 at each scheduled nonterminal level with its recorded configuration. After the root, use the source list from (54). An offloaded edge also runs Stage 3 and passes its setup opening to the immediate successor.
2. **Terminal.** The final fold produces the canonical terminal inner state. Apply the tensor prefix when required and the direct checks of Section 8.2 to this singleton state; reject any pending setup opening.

$\mathcal{V}$ accepts iff every sum-check verifies, every payload and shape check passes, the grinding nonces respect their scheduled ranges, and the terminal checks hold.

Fig. 8: The complete Akita protocol. Root multi-group batching changes only the source $\mathcal{O}_0$; setup offloading changes one producer edge and the source topology of its immediate successor. The configuration fields of (55) are recorded at each committed fold, with the opening method derived by (160).

10 Security of Akita

The object of analysis is the assembled protocol of Figure 8: the folds of Figure 5, whose per-fold checks are the rows of the relation matrix (Figure 6), closed by the terminal checks of Section 8.2, under an admissible schedule in the sense of Definition 9.2. A scheduled fold may have one or several source instances, and may replace its direct setup scan by the offloading transition of Section 9. The analysis below covers both cases. Public batching adds only the claim-combination check of Lemma 10.33.

The main theorem (Theorem 10.44) keeps two security coordinates separate. The Module-SIS inventory may target 128-bit quantum attack cost. The noninteractive knowledge-error bound is instead a schedule-specific statement in the classical random-oracle model. QROM security is outside the scope of this paper; we identify it as future work in Section 14. The bound below equals the full interactive error, including all fold, ring-reduction, sum-check, norm, batching, offloading, and terminal terms, multiplied by the random-oracle query loss. This bound takes no security credit for algebraic-challenge proof-of-work; its separate work accounting is described below. The exact field-size ceiling for this conservative bound and the requirements for a claimed Fiat–Shamir bit level appear in Corollary 10.46 and Section 12.2. The knowledge-soundness proof itself proceeds from one-fold backward extraction, to interactive extraction for a fixed schedule, and finally to the Fiat–Shamir transform.

Figure 9 summarizes the backward extraction. Each fold uses the authenticated state supplied by its successor; its range and relation checks then support the coordinate forks that recover the incoming weak openings.

Fixed-filter sampling. The concrete challenge sampler uses a finite raw-draw allowance. The next lemma records both its abort probability and its conditional output distribution.

Lemma 10.1 (Bounded sampling from a fixed accepted family). *Let $\mathcal{C}^{\mathsf{acc}} \subseteq \mathcal{C}_0$ be nonempty, let $\alpha := |\mathcal{C}^{\mathsf{acc}}|/|\mathcal{C}_0|$, and sample at most $L \geq 1$ independent uniform elements of $\mathcal{C}_0$. Output the first element in $\mathcal{C}^{\mathsf{acc}}$, or $\perp$ if all L draws are rejected. Then*

$$\Pr[\perp] = (1-\alpha)^L, \qquad \Pr[c = x \mid c \neq \perp] = \frac{1}{|\mathcal{C}^{\mathsf{acc}}|} \quad \textit{for every } x \in \mathcal{C}^{\mathsf{acc}}. \tag{177}$$

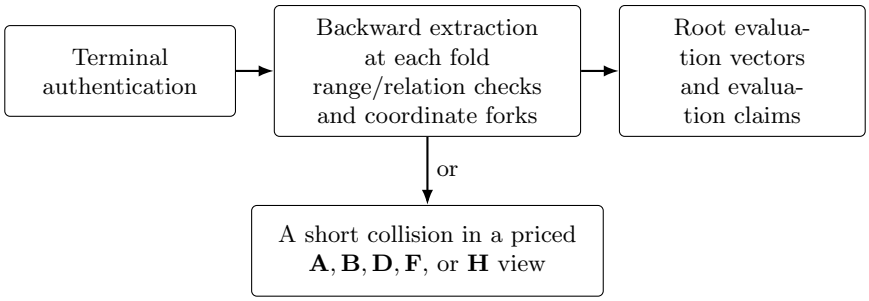

Fig. 9: The extraction argument. The terminal supplies the initial authenticated state. At each preceding fold, extraction returns incoming weak source openings or a short matrix collision. Cross multiplication establishes the source consistency needed to continue to the root relation (Theorems 10.40 and 10.44).

For independent coordinate samplers indexed by $u \in U$, the probability that all coordinates are produced is

$$a := \prod_{u \in U} \left(1 - (1 - \alpha_u)^{L_u}\right), \tag{178}$$

and, conditioned on this event, the output tuple is uniform over $\prod_{u \in U} \mathcal{C}_u^{\mathsf{acc}}$. Replacing each α_u by a certified lower bound gives a lower bound on a.

Proof. The failure probability is the probability that all raw draws lie outside the accepted set. For $x \in \mathcal{C}^{\mathsf{acc}}$, the probability that the first accepted draw equals x is

$$\sum_{j=0}^{L-1} (1-\alpha)^j \frac{1}{|\mathcal{C}_0|} = \frac{1 - (1-\alpha)^L}{|\mathcal{C}^{\mathsf{acc}}|}.$$

Dividing by the probability of producing an output proves conditional uniformity. Independence across the coordinate substreams gives (178) and the product distribution.

Lemma 10.2 (Fiat–Shamir compatibility of a fixed filter). *Fix the sampler of Lemma 10.1, and suppose each coordinate $u \in U$ is derived from a separately indexed, domain-separated ideal coordinate stream supplying its L_u independent shell-valued draws. Assume that decoding one shell draw and sampling its bit encoding conditional on a prescribed shell element both take expected polynomial work. The queries share a fixed round context but do not absorb one another's answers. Only the sampler output in $\prod_{u \in U} \mathcal{C}_u^{\mathsf{acc}}$, or $\perp$, is used as the transcript challenge. Write*

$$a_u := 1 - (1 - \alpha_u)^{L_u}.$$

For every accepted tuple $c = (c_u)_{u \in U}$, there is an exact sampler for an ideal coordinate tape conditioned on producing c. Its expected work, counting shell draws, filter tests, and conditional shell encodings, is

$$O\left(\sum_{u \in U} \frac{L_u}{a_u}\right).$$

Consequently, if membership in each accepted family is efficient and every L_u and $1/a_u$ is polynomially bounded, the classical-ROM tree-building reduction for challenges sampled uniformly from $\prod_u \mathcal{C}_u^{\mathsf{acc}}$ also applies to the fixed-filter protocol in this programmable ideal-coordinate-stream model, with polynomial overhead. Its coordinate-wise special-soundness denominators remain $|\mathcal{C}_u^{\mathsf{acc}}|$.

Proof. Consider one coordinate and fix $c \in \mathcal{C}^{\mathsf{acc}}$. Sample a uniform length-L raw tape until the bounded sampler succeeds. Let J be the position of its first accepted draw, replace that draw by c, and leave the rejected prefix and unused suffix unchanged. Before replacement, conditioned on success, the first accepted value is uniform on $\mathcal{C}^{\mathsf{acc}}$ and independent of J, the rejected prefix, and the unused suffix. The resulting

tape therefore has exactly the conditional distribution of a uniform tape given that its sampler output is c. Rejection sampling needs $1/a$ trials in expectation and examines at most L draws per trial. Applying this construction independently to every coordinate gives the claimed shell-valued tuple sampler.

To realize this tape in bits, sample the encoding of each shell element conditional on that element, then concatenate the encodings and append a fresh uniform suffix. This step is necessary: replacing a decoded shell element does not itself produce a correctly distributed bit tape. For Akita's sparse shell decoder, choose a uniform ordering of the prescribed support within each magnitude class and invert the partial Fisher–Yates shuffle to obtain its accepted bounded integers. For $n = 1$, the decoder returns zero without consuming bits. For $n > 1$, the bitmask rejection decoder samples $b = \lceil \log_2 n \rceil$ bits until the value is below n. Conditional on its accepted value, sample the geometric number of rejected words, each uniform in $[n, 2^b)$, followed by that value; unused bits of the underlying byte words remain uniform. Condition each sign byte on its required low bit and sample its remaining bits uniformly. This gives the exact conditional encoding. Integer rejection uses fewer than two word attempts in expectation; the shell's bounded support size gives expected polynomial decoding work. The displayed cost counts shell operations, each with this expected polynomial bit cost.

The implementation bounds the outer fixed-filter search by 4096 completed shell draws but does not limit the inner integer rejection. In the ideal-stream model that rejection loop terminates almost surely; there is no separate bounded-decoder exhaustion event. The conditional construction programs that ideal stream, not a preimage of the concrete 256-bit SHAKE seed.

In the classical-ROM programming step, program the challenged coordinate's oracle input with its conditional tape and keep all other coordinate inputs and answers fixed. Conditioned on the chosen accepted challenge, the adversary sees the same oracle-answer distribution as in the real fixed-filter protocol. Hence the direct accepted-family reduction lifts without changing its counting denominator; only the conditional-tape sampling work is added.

Field vectors returned together. The tensor-row, range-point, and relation-row challenges are uniform field vectors. Their extraction uses Lemma 10.14, not the flat sibling structure of Definition 3.11. At a rewind, keep the selected oracle input and the preceding transcript fixed, program the required coordinate prefix of its answer, and draw its remaining coordinates independently. The ideal Fiat–Shamir input encodes the stage and complete preceding transcript, so matching inputs preserve the prefix required by that lemma. The prover receives the complete vector, exactly as in the protocol; the rewind does not replace one vector query by several scalar queries.

In the ideal-XOF model, condition the answer tape on the prescribed field coordinates. For uniform base-field rejection decoding, sample the rejected words and the accepted word conditional on its prescribed value, as in the bit-decoding argument above; use independent coordinates for an extension-field element and leave the unused tape suffix uniform. This is an exact polynomial-time conditional sampler. Nested extraction programs these ideal answers, not the concrete seed from which the implementation expands its field challenges. Together with Lemma 10.2, this supplies the conditional-answer interface for both kinds of stage in Lemma 10.14. The Fiat–Shamir extractor of Theorem 10.44 needs this interface only at fold challenges. Every other stage is a checkpoint or plain stage of Proposition 10.43, and the extractor never programs a prefix of its answer. The prefix sampler above is used only for the complete transcript tree of Theorem 10.40.

A response-dependent nonce grind is different from this fixed filter. The schedule records the nonce range, its cardinality, and the deterministic retry rule, but the set of nonces accepted by that rule depends on the witness. The grind therefore does not define a smaller fixed set $\mathcal{C}^{\mathsf{acc}}$. Every Fiat–Shamir-namespace probe used to search the nonce range is instead included in Q. The schedule-wide claim is restricted to $Q \le Q_{\max}$, so a parameter checker must ensure that the allowed honest grind fits within that public query budget. The checker must not both shrink $|\mathcal{C}^{\mathsf{acc}}|$ for the grind and charge the same probes through $Q_{\max}$.

Algebraic-challenge proof-of-work. Akita uses transcript-bound proof-of-work before selected algebraic challenges, independently of response retries. At such a site, L is the numerator of an algebraic error bound, such as a polynomial degree or a sum of degree bounds. For an error bounded by $L2^{-C}$, the rule $g = \max\{0, 128 + \lceil \log_2 L \rceil - C\}$ makes $2^{-g} L 2^{-C} \le 2^{-128}$. The implementation uses the challenge field's nominal capacity C in this rule; its production profiles use $C = 128$. The analysis below uses the actual bad-challenge fraction ϵ (for example, $L/|E|$), so it does not identify a nominal C-bit field with a field of exactly 2^C elements.

The public plan fixes each protected site, its target g, and its nonce width. 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 output. A zero-bit site is a no-op. The instance descriptor binds the plan; verification replays the exact order and widths of the packed nonce stream. This mechanism applies to standalone PCS proofs and the Jolt backend (Appendix E.5).

Proposition 10.3 (Adaptive search at a protected challenge). *Fix a protected site with target g. A candidate context binds the transcript prefix, site, and nonce to one predicate/challenge pair. Distinct contexts have distinct independent pairs of ideal oracle outputs. For each context c, let B_c be its bad-challenge set. Assume this set is fixed before either output of that pair is first queried and, conditional on the preceding history, has mass at most ϵ under the challenge distribution. The predicate passes with probability 2^{-g} independently of that challenge.*

An adaptive prover that queries either output of at most N distinct candidate pairs and finally submits one candidate has probability at most

$$\min\{1, (N+1)2^{-g}\epsilon\}$$

of submitting a passing predicate with a challenge in its bad set. Queries may occur in either order, including before the candidate is used in the proof. The same statement holds for $g = 0$ with a vacuous predicate.

Proof. Enumerate candidates when either output is first queried. Conceptually sample both outputs at that time, revealing only those actually queried. Conditioned on the history before this first query, the context and its bad set are fixed and the two fresh outputs are independent. Thus the probability that this pair both passes and is bad is at most $2^{-g}\epsilon$. Later queries and the decision whether to use the pair cannot enlarge this event. A union bound over the at most N first-touched pairs gives $N2^{-g}\epsilon$. If the final candidate was never touched, its verification exposes one additional fresh pair with the same bound. If either output was already queried, that candidate was already counted. This argument also covers stopping early and choosing later contexts from previous answers.

Corollary 10.4 (Several protected and unprotected sites). *For each site s in a fixed transcript plan, suppose the hypotheses of Proposition 10.3 hold with bad fraction ϵ_s, target g_s, and at most N_s first-touched candidate pairs. Put $g_s = 0$ for an unprotected site. If the proof submits one candidate per site, then*

$$\Pr[\exists s : \text{ site } s \text{ passes with a bad challenge}] \le \min\left\{1, \sum_s (N_s+1)2^{-g_s}\epsilon_s\right\}.$$

No independence between sites is required: the per-site bounds hold conditionally on their preceding histories, and then a union bound applies. The site index includes each occurrence in the recursive schedule.

The corollary bounds adaptive search for the specified algebraic bad events, including challenge-first prequeries. In particular, success probability $\delta > 0$ for a protected event requires $N + 1 \ge \delta\, 2^g/\epsilon$. The knowledge-soundness theorem below has two kinds of error term. A fold term bounds the failure to collect sibling transcripts. This failure event is not membership in a bad set fixed before the first query of a candidate context, so Corollary 10.4 does not rescale it. Every other term belongs to a checkpoint or plain stage of Proposition 10.43. Its bad set is determined by the stage input and the fixed witness candidate of its checkpoint, so the hypotheses of Proposition 10.3 hold and a protected stage enters the refined bound (234) with the factor 2^{-g} (Remark 10.17). The budget (226) does not use this factor.

10.1 Completeness

The algebraic identities of an honest execution hold exactly, but the execution can still abort. At each response-producing stage, a bounded challenge sampler may fail, or the resulting response may lie outside its scheduled encoding or norm bound. We first express the probability of these events for one stage and then account for them across the schedule.

LaBRADOR and Greyhound also use distributional estimates to choose concrete response bounds [18, §5.4][81, §5]. Here we make the required premise explicit for every preceding honest transcript and account for finite response retries, bounded challenge sampling, zero tensor factors, and grinding exhaustion throughout

the recursion. Proved response bounds establish this premise unconditionally; our optimized later-fold bounds use the response-model premise in Equation (358).

Let $\mathsf{RespLev}(\mathsf{sch})$ be the set of response-producing stages: every nonterminal fold and the Akita terminal. Fix a stage $h \in \mathsf{RespLev}(\mathsf{sch})$ and an honest transcript prefix τ immediately before its response nonce is sampled. This prefix fixes every source vector used at stage h. We shall define $q_h(\tau)$ as the conditional probability that one candidate nonce is unusable. Since all $N_{\mathsf{try},h}$ retries use the same fixed sources, we first bound this probability for each τ and only then average over honest execution histories.

Response encoding and norm bounds. Fix an admissible schedule sch and a response-producing stage $h \in \mathsf{RespLev}(\mathsf{sch})$. Let $\mathcal{R}_h$ index the folded responses checked at that stage: one element for the core relation, one element per source instance for the multi-instance relation, and one element at the terminal. The response-chunked specialization is given in Corollary 11.3.

For response r, let $\mathsf{Enc}_{h,r}(\mathbf{z})$ be membership in its exact scheduled encoding domain, evaluated on centered integer coefficients. The schedule records a symmetric threshold $T_{h,r}$ whose coefficient cube is contained in this domain. At a nonterminal fold using balanced positional base $b_{h,r}$ and depth $\delta_{f,h,r}$, the exact domain is the interval below in every coefficient:

$$M_{h,r} := \frac{b_{h,r}}{2}\frac{b_{h,r}^{\delta_{f,h,r}}-1}{b_{h,r}-1}, \qquad T_{h,r} := \left(\frac{b_{h,r}}{2}-1\right)\frac{b_{h,r}^{\delta_{f,h,r}}-1}{b_{h,r}-1}. \tag{179}$$

Thus $\mathsf{Enc}_{h,r}(\mathbf{z})$ is exactly the event that every coefficient lies in $[-M_{h,r}, T_{h,r}]$, or equivalently that the canonical decomposition fits in $\delta_{f,h,r}$ planes. At the terminal it is the exact coefficient domain accepted by the scheduled clear prefix or fixed-width encoding. If response r uses the Euclidean route, let $S_{\max,h,r}$ be its scheduled complete response squared-norm bound.

Definition 10.5 (Honest response admission). *With these encoding domains and norm bounds, define*

$$\mathsf{HAdm}_{h,r}(\mathbf{z}) := \mathsf{Enc}_{h,r}(\mathbf{z}) \wedge \begin{cases} \mathsf{true}, & p_{h,r} = \infty, \\ \|\mathbf{z}\|_{2,\mathrm{coef}}^2 \le S_{\max,h,r}, & p_{h,r} = 2. \end{cases}$$

Failure of one nonce attempt. Let $\mathsf{Hist}_h^{\mathsf{hon}}$ be the set of reachable honest transcript prefixes immediately before the nonce for stage h is sampled. At an evaluation-trace receiver, the 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 all source vectors.

Definition 10.6 (Conditional nonce failure). *Let U_h index the fold-challenge coordinates derived from one candidate nonce. Coordinate u samples from a raw shell $\mathcal{C}_{0,h,u}$ into the accepted family $\mathcal{C}_{h,u}^{\mathsf{acc}}$ with raw-draw allowance $L_{h,u}$. Write $\alpha_{h,u} := |\mathcal{C}_{h,u}^{\mathsf{acc}}|/|\mathcal{C}_{0,h,u}|$ and define the challenge-construction success probability*

$$a_h := \prod_{u \in U_h} \left(1 - (1-\alpha_{h,u})^{L_{h,u}}\right). \tag{180}$$

An unfiltered coordinate has $\alpha_{h,u} = 1$. A fresh candidate nonce produces either $\perp$ or a tuple $\boldsymbol{c}_h$ that, conditioned on not being $\perp$, is uniform over $\prod_{u\in U_h} \mathcal{C}_{h,u}^{\mathsf{acc}}$ by Lemma 10.1. For a successful tuple, define

$$q_h^{\mathsf{rsp}}(\tau) := \Pr_{\boldsymbol{c}_h}\left[\neg \bigwedge_{r\in\mathcal{R}_h} \mathsf{HAdm}_{h,r}(\mathbf{z}_{h,r}(\tau,\boldsymbol{c}_h)) \;\middle|\; \boldsymbol{c}_h \neq \perp\right], \tag{181}$$

$$q_h(\tau) := 1 - a_h\big(1 - q_h^{\mathsf{rsp}}(\tau)\big). \tag{182}$$

Thus $q_h(\tau)$ is the probability that the complete nonce attempt fails, including challenge-sampler exhaustion.

At a nonterminal fold the range proof may certify a wider alphabet than the honest base-$b_{h,r}$ alphabet. Every accepted response is therefore bounded for security by $\beta_{\mathsf{fold}}^{\mathsf{cert}}$ from (109), and two accepted responses differ by at most Δ_f^{cert} from (110). Neither quantity replaces the honest encodability event in Definition 10.5.

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 samples from the same bounded challenge-sampling procedure, while the prefix τ and hence every source vector remain fixed. The probability that all trials fail is therefore the stated power. Once an admissible nonce is found, the canonical response decompositions exist at a nonterminal fold, the terminal encoding is valid at the terminal, and every scheduled Euclidean bound is satisfied. 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. The terminal checks are likewise exact identities on its revealed honest source.

The one-stage lemma reduces schedule completeness to a question about the conditional quantities $q_h(\tau)$. We record the pointwise bound that will let us answer this question uniformly over transcript histories.

Definition 10.8 (Uniform conditional nonce-failure bound). *For numbers $(\delta_h)_{h\in\mathsf{RespLev}(\mathsf{sch})}$, we say that the schedule has a uniform conditional nonce-failure bound $(\delta_h)_h$ if*

$$q_h(\tau) \le \delta_h \qquad \text{for every } h \in \mathsf{RespLev}(\mathsf{sch}) \text{ and } \tau \in \mathsf{Hist}_h^{\mathsf{hon}}. \tag{183}$$

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*

$$q_h^{\mathsf{rsp}}(\tau) \le \widehat{\delta}_h^{\mathsf{rsp}} \qquad \text{for every } h \in \mathsf{RespLev}(\mathsf{sch}) \text{ and } \tau \in \mathsf{Hist}_h^{\mathsf{hon}}. \tag{184}$$

The response model of Appendix G.3 supplies the bounds and proposed values $\widehat{\delta}_h^{\mathsf{rsp}}$. Its Gaussian surrogate motivates (184), but does not prove it for the actual integer response. Challenge-sampler success remains a separate, combinatorially certified quantity.

The conditioning in (183) is essential. An average one-attempt bound

$$\mathbb{E}_\tau[q_h(\tau)] \le \delta_h$$

does not in general imply

$$\mathbb{E}_\tau[q_h(\tau)^{N_{\mathsf{try},h}}] \le \delta_h^{N_{\mathsf{try},h}}.$$

The retries reuse the source vectors fixed by τ, so the power must be taken before averaging over τ.

We can now state the schedule-level bound. Besides response-nonce exhaustion, it charges two further events: a required final factor of an evaluation-trace tensor reduction vanishes, or a proof-of-work search exhausts its nonce allowance.

For each active proof-of-work site i in the schedule, let $g_i > 0$ be its target and M_i its nonce allowance. Define

$$\varepsilon_{\mathsf{pow}}(\mathsf{sch}) := \sum_{i:g_i>0} (1 - 2^{-g_i})^{M_i}. \tag{185}$$

The implementation uses $M_i = 2^{g_i+7}$, so each summand is at most e^{-128}. A zero-bit site performs no search and cannot exhaust.

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*

$$\mathbb{E}\big[\mathbf{1}_{\{\mathsf{Reach}_h\}} q_h(\mathsf{Tr}_{<h})^{N_{\mathsf{try},h}}\big] \le \xi_h \tag{186}$$

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 each* h *in this set, let* G_h *be its number of incoming groups,* κ_h *its tensor split parameter, and* $m_{\max,h}$ *its largest tensor-reduction tail arity; write* E_h *for its extension field. Then the honest prover is accepted in the random-oracle model with probability at least*

$$1 - \varepsilon_{\mathsf{comp}}(\mathsf{sch}),$$

$$\varepsilon_{\mathsf{comp}}(\mathsf{sch}) := \sum_{h \in \mathsf{RespLev}(\mathsf{sch})} \xi_h + \sum_{h \in \mathsf{EvalRecv}^{+}(\mathsf{sch})} \frac{G_h(\kappa_h + m_{\max,h})}{|E_h|} + \varepsilon_{\mathsf{pow}}(\mathsf{sch}). \tag{187}$$

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 expectation on the left-hand side of (186). 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 (187).

Corollary 10.10 (Conditional completeness). *Suppose* $q_h(\tau) \le \delta_h$ *for every* $h \in \mathsf{RespLev}(\mathsf{sch})$ *and every reachable honest prefix* $\tau \in \mathsf{Hist}_h^{\mathsf{hon}}$*. Then Theorem 10.9 applies with*

$$\xi_h = \delta_h^{N_{\mathsf{try},h}}.$$

In particular, the schedule's completeness error is bounded by

$$\varepsilon_{\mathsf{comp}}(\mathsf{sch}) \le \sum_{h \in \mathsf{RespLev}(\mathsf{sch})} \delta_h^{N_{\mathsf{try},h}} + \sum_{h \in \mathsf{EvalRecv}^{+}(\mathsf{sch})} \frac{G_h(\kappa_h + m_{\max,h})}{|E_h|} + \varepsilon_{\mathsf{pow}}(\mathsf{sch}). \tag{188}$$

This is an unconditional completeness bound when each sampler-success bound and response-admission bound is proved. If any response bound is instead assumed from the response model, the conclusion is conditional on the corresponding modeled-response assumption.

Proof. For every reachable prefix, $q_h(\mathsf{Tr}_{<h})^{N_{\mathsf{try},h}} \le \delta_h^{N_{\mathsf{try},h}}$. Substitute this inequality into (186).

Allowing rare exceptional histories. Uniform control of $q_h(\tau)$ is sufficient, but not necessary. Suppose a proved set of good prefixes $\mathcal{G}_h$ satisfies $\Pr[\mathsf{Reach}_h \wedge \mathsf{Tr}_{<h} \notin \mathcal{G}_h] \le \eta_h$ and $q_h(\tau) \le \delta_h$ on $\mathcal{G}_h$. Splitting the expectation in (186) according to whether the prefix lies in $\mathcal{G}_h$ shows that one may take

$$\xi_h = \eta_h + \delta_h^{N_{\mathsf{try},h}}.$$

This lets a proved high-probability invariant on recursive source squared norms replace a deterministic one. The exceptional probability η_h is charged once rather than raised to the retry count, since retries hold the source history fixed.

Obtaining the response-admission bound. The completeness argument now reduces to bounding $q_h(\tau)$ for the sources fixed by each honest prefix. There are two certified ways to supply this bound. Deterministic coefficient and squared-norm envelopes can make every successfully sampled response fit its scheduled bounds. Alternatively, tail bounds can allocate a failure probability to each response's encoding and squared-norm check; a union bound combines these probabilities without assuming independence between the checks or between responses. Appendix G.2 gives both constructions.

The required source estimates depend on the challenge distribution. Appendix G.1 treats dense and canonical one-hot sources under the stated unfiltered sampling rules. Appendix G.1 treats the actual accepted distribution of a symmetric fixed filter. Filtering need not preserve the independent signs used by an unfiltered

tail bound, so that bound cannot be reused without checking its premises. All source envelopes must hold after fixing the preceding transcript, including its earlier successful nonce searches.

When a schedule chooses its response bounds from the Gaussian surrogate instead, the same framework yields a conditional statement. If the model-derived bounds satisfy (184), then (182) bounds the complete per-attempt failure probability by

$$\widehat{\delta}_h^{\mathsf{tot}} := 1 - a_h(1 - \widehat{\delta}_h^{\mathsf{rsp}}). \tag{189}$$

Here a_h is determined by the actual accepted-family density and raw sampling allowance; a certified lower bound on a_h gives the corresponding upper bound on $\widehat{\delta}_h^{\mathsf{tot}}$. Applying Corollary 10.10 with $\delta_h = \widehat{\delta}_h^{\mathsf{tot}}$ gives the conditional completeness error in (188). The modeled premise affects only honest abort probability and expected prover work, not the knowledge-soundness theorem below.

Choosing the first valid proof-of-work predicate before drawing its independent algebraic challenge preserves the challenge distributions used above. Its exhaustion probability is already included in (187).

10.2 Knowledge Soundness

Transcript Trees and Checkpoint Extraction The results of this subsection do not depend on Akita's algebra. They extend the coordinate-wise special-soundness framework of Section 3.8 in three steps. Corollary 10.11 allows the coordinates of one challenge to come from sets of different sizes. Lemma 10.14 allows a vector challenge to be interpolated one coordinate at a time. Theorem 10.16 removes the rewinding at every stage that only tests a witness which the transcript has already fixed.

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*

$$\sum_i \sum_{u\in U_i} \frac{k_{i,u}-1}{|\mathcal{X}_{i,u}|}, \qquad \prod_i \left(1 + \sum_{u\in U_i} (k_{i,u}-1)\right), \tag{190}$$

respectively. The Fiat–Shamir transform is knowledge sound against a classical Q-query random-oracle adversary with the first quantity multiplied by $Q+1$.

The homogeneous per-round case has $|U_i| = \ell_i$, $\mathcal{X}_{i,u} = \mathcal{X}_i$, and $k_{i,u} = k_i$. It therefore contributes $\ell_i(k_i - 1)/|\mathcal{X}_i|$ to the error and $\ell_i(k_i-1)+1$ to the tree product, as noted in [42, 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.

Proof. Run the counting argument of Theorem 3.13 separately for each coordinate. Failure to obtain the $k_{i,u}$ required values contributes at most $(k_{i,u}-1)/|\mathcal{X}_{i,u}|$. A union bound over the coordinates and rounds gives the first expression. At round i, the special-sound structure has $1 + \sum_u (k_{i,u}-1)$ children, giving the corresponding factor in the second expression. The classical Fiat–Shamir statement follows by applying the same $Q+1$ loss to the resulting interactive error.

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 3.11.*

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, put*

$$e_r := \sum_{u \in U_r} \frac{k_{r,u} - 1}{|S_{r,u}|}, \qquad b_r := \begin{cases} 1 + \sum_{u \in U_r} (k_{r,u} - 1), & \textit{flat stage,} \\ \prod_{u \in U_r} k_{r,u}, & \textit{nested stage.} \end{cases}$$

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 its complete preceding transcript, including the earlier challenges. Second, an oracle answer can be sampled conditional on any prescribed prefix of its challenge coordinates in expected polynomial time.*

The first condition implies that matching stage inputs have matching transcript prefixes. The second concerns the full answer distribution, including any sampler encoding; it does not ask for a preimage of a concrete hash seed. A vector challenge returned by one oracle query counts as one query in the bound.

Proof. Flat stages use Corollary 10.11. For a nested stage, perform scalar tree sampling recursively, starting with the last coordinate. The successful outputs at coordinate u are complete suffix trees for coordinates $u + 1, \ldots, a$, followed by the required later-round trees. Collect k_u distinct values of coordinate u with such outputs, keeping the earlier coordinates fixed. Their suffix trees may differ. This gives exactly Definition 10.12, rather than coordinate pairs around a single vector.

The scalar sampling argument charges $(k_u - 1)/|S_u|$ at this step and multiplies the continuation-tree size by k_u. One way to see its counting threshold is to prune a product set from its last coordinate upward: a prefix with fewer than k_u surviving children loses at most $(k_u - 1)/|S_u|$ of the uniform conditional mass. A union bound over the coordinates gives e_r. The weighted scalar sampling argument handles the cost of obtaining the surviving continuations; applying it recursively gives the product of the k_u factors, with polynomial sampling overhead.

For the random-oracle statement, we give the recursion underlying the application of [42, Section 8, Lemmas 8.2–8.3]. Fix the prover's random tape. Let J be the random oracle table, whose entries are complete answers, and let $I_r(J)$ be the query input selected by the original execution for stage r. If that execution does not reach stage r, use a failure symbol. Every recursive procedure returns this original index, even when its suffix extraction fails; it must not replace it with an index selected on a successful rerun. Its success output also records the original pre-challenge transcript and a tree rooted there. Equality of selected stage inputs fixes that prefix by hypothesis, so requiring it introduces no additional rejection event.

Fix the suffix procedure's auxiliary sampling tapes independently of J, while retaining fresh randomness for the enclosing scalar sampling game. These fixed tapes make the suffix procedure a deterministic table map $M(J) = (V, I_r(J))$, with a tree attached when $V = 1$. Average over these tapes after applying the sampling bounds. For a candidate table, first run the original execution to determine I_r and its prefix. An index or prefix mismatch is a cheap failure; invoke the costly suffix procedure only for a matching candidate. This is the matching-index/nonmatching-index cost split required by the weighted sampling game. Within a vector, the recursion programs only the selected table entry; its successive suffix procedures fix the prescribed coordinate prefix and may choose different remaining coordinates.

For an entry i, write J_{-i} for all 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 is in this original-index support. Hence every support quantity in the recursive sampling games is at most $P_{0,r}$. Moreover $P_{0,r} \leq Q + 1$: replacing entry i cannot change an

execution before its first query to i. If that input is never queried in the original execution, a replacement leaves the execution unchanged, so it can select i only if i was already the output challenge input. Pointwise, there are at most Q queried inputs and one such unqueried output input. The same argument applies after fixing any auxiliary tapes; the support uses the original execution, not all the queries made by the extractor's reruns.

Let s_{u+1} and C_{u+1} denote success probability and expected cost before adding coordinate u to a nested stage. Apply the scalar game with arity k_u to the deterministic suffix map above, retaining its original index also on failure. Its success bound and weighted cost bound give

$$s_u \geq s_{u+1} - P_{0,r}\frac{k_u - 1}{|S_u|}, \qquad C_u \leq k_u C_{u+1} + (k_u - 1)P_{0,r}\gamma_u,$$

where γ_u bounds the cheap original-execution and conditional-answer sampling work. Sampling costs can be charged in expectation by conditioning on their tapes and then averaging. The cost of a matching candidate includes its complete suffix extraction. Thus this recurrence does not charge every failed candidate for an expensive suffix computation. 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. There are no virtual protocol messages or extra adversarial oracle queries.

Finally, induct backward over the actual protocol stages. The base procedure checks an accepting transcript. At stage r, its continuation procedure returns complete trees for all later stages and keeps the original I_r and prefix even on failure. Later-stage programming occurs only within these continuations. A matching later-stage input binds the entire transcript before that stage, so it also preserves every earlier prefix. A flat stage uses the flat weighted game, and a nested stage uses the coordinate recursion just proved. Writing s_{r+1}, C_{r+1} for the continuation's probability and cost, the resulting bounds have

$$s_r \geq s_{r+1} - (Q+1)e_r, \qquad C_r \leq b_r C_{r+1} + (b_r - 1)(Q+1)\gamma_r$$

for a polynomial bound γ_r on the stage's cheap work. This also allows early rejection of a changed earlier prefix. The success losses therefore add over stages, rather than acquiring a new factor $Q+1$ at every depth. The cost is polynomial in Q, the protocol description, and $\prod_r b_r$. Applying the assumed deterministic tree extractor completes the proof. In the interactive case there is one fixed challenge node instead of an adversarially selected oracle input, so the same recurrences use $P_{0,r} = 1$.

Extraction with checkpoints. Lemma 10.14 branches the transcript tree at every stage. A protocol that commits to a witness and then tests it needs less. Once a candidate witness has been extracted from the transcripts below the commitment, the tests are public-coin checks of one fixed object. A false claim about a fixed object passes a challenge only if the challenge lies in a small set, its *bad set*, which is determined before the challenge is drawn. The extractor therefore needs no sibling transcripts at the tests. It evaluates the tested relation on the candidate. If the relation holds, nothing further is required. If it fails, one more extraction below the same commitment either returns a different candidate, which breaks binding, or returns the same failing candidate, which requires a challenge in its bad set. The tree then branches at the stages whose algebra needs several transcripts, and it at most doubles at each test. This is the commitment-predicate argument of Aardal et al. [1, Section 5 and Appendix H], with two changes. Their single predicate challenge becomes a sequence of challenge stages, which we analyse round by round [30] against the fixed candidate. The argument is carried out inside the multi-round sampling game of [12].

We use the transcript conventions of Lemma 10.14. A protocol has stages $1, \ldots, m$ and transcripts $(a_1, \rho_1, \ldots, a_m, \rho_m, a_{m+1})$, where a_1 contains the statement. The stage-r input u_r injectively encodes r and the transcript through a_r, for $1 \leq r \leq m+1$; thus u_{m+1} encodes the complete transcript. For $r \leq m$, ρ_r is the complete oracle answer at u_r, distributed as Ω_r; it includes every coordinate of a vector challenge and any proof-of-work predicate bits that the verifier checks at that stage (Section 10). We write $u \preceq u'$ when the transcript encoded by u' extends the one encoded by u, and $\mathrm{Sub}(u) := \{u' : u \preceq u'\}$. For a prover P^* with a fixed random tape and an oracle table J, let A^J run P^* and then the verifier, and let $u_r(J)$ be the stage-r input of its output transcript.

Definition 10.15 (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 and computable in polynomial time.*

(i) Stage types. *A partition* $\{1,\dots,m\} = X \sqcup H \sqcup N$ *into extraction, checkpoint, and plain stages. The* segment *of* $h \in H$ *is* $\mathrm{seg}(h) = \{h,\dots,\bar{h}\}$*, where* $\bar{h}+1$ *is the next stage in* $X \cup H$*, or* $\bar{h} = m$*. 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*.*

(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;* $u_j(y)$ *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* u *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* u*, or an element of* $\mathcal{T}$*.*

(v) Checkpoint stages. *For* $h \in H$*, a candidate space* W_h*, maps* $\mathsf{Cand}_h : Y_{h+1} \to W_h$ *and* $\mathsf{Good}_h : Y_{h+1} \to \{0,1\}$*, and* $Y_h := \{y \in Y_{h+1} : \mathsf{Good}_h(y) = 1\}$*.*

(vi) Binding. *For* $h \in H$ *and* $y, y' \in Y_{h+1}$ *with* $u_h(y) = u_h(y')$ *and* $\mathsf{Cand}_h(y) \ne \mathsf{Cand}_h(y')$*, a map* Coll_h *returns* $\mathsf{Coll}_h(y,y') \in \mathcal{T}$*.*

(vii) Segment soundness for a fixed candidate. *For* $h \in H$*,* $w \in W_h$*,* $j \in \{h,\dots,\bar{h}+1\}$*, and every stage-*$j$ *input* u*, a bit* $D(u;w)$*; and for* $j \le \bar{h}$ *a set* $\mathsf{Bad}_j(u;w) \subseteq \operatorname{supp} \Omega_j$ *of probability at most* ε_j *under* Ω_j*. They satisfy:*

(a) *if* $y \in Y_{h+1}$ *and* $\mathsf{Good}_h(y) = 0$*, then* $D(u_h(y); \mathsf{Cand}_h(y)) = 1$*;*

(b) *if* $D(u;w) = 1$ *for a stage-*j *input* u *with* $j \le \bar{h}$ *and* $\rho \notin \mathsf{Bad}_j(u;w)$*, then* $D(u';w) = 1$ *for every stage-*$(j+1)$ *input* u' *that extends* u *by the answer* ρ*;*

(c) *if* $y \in Y_{h+1}$*, then* $D(u_{\bar{h}+1}(y); \mathsf{Cand}_h(y)) = 0$*.*

(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* $\widehat{K} := 2^{|H|} \prod_{r \in X} b_r$*.*

The items follow the extractor. A certificate in Y_r is what the extractor holds after it has processed stages $r,\dots,m$: one transcript prefix, together with what the later extraction stages have recovered for it, such that every checkpoint at stage r or later has passed its test. At a plain stage the extractor does nothing, which is item (iii). At an extraction stage it combines b_r certificates into one; special soundness of the stage is the requirement in item (iv) that Comb_r be defined on every such family. At a checkpoint it reads the candidate $\mathsf{Cand}_h(y)$ from a certificate and evaluates the test $\mathsf{Good}_h(y)$, which must therefore be computable. Item (vi) is binding: two certificates with the same checkpoint input and different candidates yield a witness of $\mathsf{R}_{\mathsf{bind}}$. Item (viii) turns a certificate at the first stage into a witness of R.

Item (vii) treats the segment as a public-coin proof about one fixed candidate w. The bit $D(u;w) = 1$ records that the claims made in the transcript prefix u are false for w. Condition (a) says that a failed test makes the claims at the checkpoint input false. Condition (b) is round-by-round soundness: false claims remain false unless the answer lies in the bad set. Condition (c) says that the claims at the end of the segment are true for the candidate of every certificate. Together they give the fact used in the proof: if a certificate fails the test, then some answer on its path through the segment lies in a bad set of its candidate. The factor $2^{|H|}$ in the tree size accounts for the one additional extraction that a checkpoint may perform.

Theorem 10.16 (Checkpoint extraction). *Assume a checkpoint structure whose stages in* X *satisfy the conditional-answer sampling hypothesis of Lemma 10.14. Let* P^* *make at most* Q *classical random-oracle queries, of which* Q_r *are to stage-*r *inputs, and output an accepting proof with probability* ε*. There is an extractor that outputs a witness for* R *or for* $\mathsf{R}_{\mathsf{bind}}$ *with probability at least*

$$\varepsilon - (Q+1)\sum_{r \in X} e_r - \sum_{r \in H \cup N} (\mathbb{E}[Q_r] + 1)\,\varepsilon_r \tag{191}$$

and runs P^* *at most* $\widehat{K} + (Q+1)(\widehat{K}-1)$ *times in expectation, with polynomial additional work per run. In particular the knowledge error relative to* $\mathsf{R} \cup \mathsf{R}_{\mathsf{bind}}$ *is at most*

$$(Q+1)\sum_{r \in X} e_r + Q \max_{r \in H \cup N} \varepsilon_r + \sum_{r \in H \cup N} \varepsilon_r \;\le\; (Q+1)\Big(\sum_{r \in X} e_r + \sum_{r \in H \cup N} \varepsilon_r\Big), \tag{192}$$

where the maximum is zero if $H \cup N$ is empty. For the interactive protocol the same statements hold with $Q = 0$.

Proof. The extractor is a recursion with one procedure E_r per stage, where E_r calls E_{r+1} once or several times; we call E_r a *node* and its calls to E_{r+1} its *children*. We first define the procedures, then state three invariants, and then bound the success probability s_r and the expected number of runs C_r of each E_r by recurrences in r. Plain and extraction stages reuse the recurrences of Lemma 10.14. The new case is a checkpoint stage, which Claims 1 and 2 below handle.

Define procedures E_r, $1 \le r \le m+1$, each returning a triple (I, v, y). The procedure $E_{m+1}(J)$ runs A^J and returns the tuple of stage inputs $I = (u_r(J))_r$, which we call the *index tuple* (its entries are the original indices $I_r(J)$ of Lemma 10.14), the verifier's decision v, and the transcript y; a failing output omits y. Elements of $\mathcal{T}$ are treated as tagged outputs without a transcript: a node that receives one from a successful child returns it as its own successful output. For $r \in N$ put $E_r := E_{r+1}$. For $r \in X$, E_r is the stage-r procedure from the proof of Lemma 10.14 (its flat or nested sampling game) with continuation E_{r+1}, followed by Comb_r on the collected children. That procedure tests whether a resampled table selects the same stage-r input before it runs the continuation in full; here the test is the first run of A inside E_{r+1}, so it needs no separate run. For $h \in H$, $E_h(J)$ proceeds as follows.

1. Run $(I, v, y) \leftarrow E_{h+1}(J)$. If $v = 0$, return $(I, 0)$.
2. If $y \in \mathcal{T}$ or $\mathsf{Good}_h(y) = 1$, return $(I, 1, y)$.
3. Otherwise put $i := u_h(J)$ and $w^0 := \mathsf{Cand}_h(y)$. Repeat the following until an attempt ends with $v' = 1$ and $u_h(J') = i$. Draw a fresh table K' on $\mathrm{Sub}(i)$, let J' agree with J outside $\mathrm{Sub}(i)$ and with K' on it, and start $E_{h+1}(J')$ with fresh coins. Stop it after its first run of $A^{J'}$ if $u_h(J') \neq i$ or $A^{J'}$ rejects; otherwise let it finish with output (I', v', y').
4. If $y' \in \mathcal{T}$, return $(I, 1, y')$. If $\mathsf{Cand}_h(y') \neq w^0$, return $(I, 1, \mathsf{Coll}_h(y, y'))$. 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 the whole table below the checkpoint input, not one entry, and step 4 reports failure when the two extractions agree on a candidate that fails the test: a failing candidate is never passed to the node above.

Invariants. By backward induction on r: (1) the index tuple returned by $E_r(J)$ is that of A^J, whether or not $v = 1$; (2) if $v = 1$ then $y \in Y_r^+$, and if $y \notin \mathcal{T}$ 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 hold by 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 input matches, together with the totality of Comb_r. For $h \in H$, step 2 returns a certificate with $\mathsf{Good}_h = 1$ or an element of $\mathcal{T}$, and step 4 applies Coll_h to $y, y' \in Y_{h+1}$ with common stage-h input i. 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, inside its checkpoints, fresh tables. As in the proof of that lemma, we fix the tapes before applying the sampling bounds and average over them afterwards. Here the tape may depend on the table: fix a function F from tables to tapes, let the invocation on table J use $F(J)$, analyse for fixed F, and average over a uniform F at the end. For a 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 by the checkpoints inside it are part of its tape. An enclosing extraction stage samples without replacement, so its invocations visit distinct tables and receive independent tapes.

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. For $r \in N$, $s_r = s_{r+1}$ and $C_r = C_{r+1}$. For $r \in X$, the proof of Lemma 10.14 gives

$$s_r \ge s_{r+1} - (Q+1)e_r, \qquad C_r \le b_r C_{r+1} + (b_r - 1)(Q+1),$$

because it uses only the properties of the continuation listed in the previous paragraph; its cheap work per resampled table is one run of A.

Checkpoint stages. Fix $h \in H$. For a stage-h input i, write a table as $J = (c, K)$, where K is its restriction to $\mathrm{Sub}(i)$ and c its restriction to the complement. For fixed (i, c), with K fresh and fresh coins, define

$$\begin{aligned} P_{i,c} &:= \Pr[E_{h+1}(c, K) \text{ returns } v = 1 \text{ and } u_h(c, K) = i], \\ q_{i,c}(w) &:= \Pr[\text{the same, and } y \notin \mathcal{T},\ \mathsf{Cand}_h(y) = w], \\ f_{i,c}(w) &:= \Pr[\text{the same as } q_{i,c}(w),\ \text{and } \mathsf{Good}_h(y) = 0]. \end{aligned}$$

Thus $P_{i,c}$ is the probability that a fresh extraction below i succeeds and selects i, $q_{i,c}(w)$ the probability that it does so with candidate w, and $f_{i,c}(w)$ the probability that this certificate also fails the test. For $j \in \mathrm{seg}(h)$, let $N_{i,j}(c, K)$ be the number of stage-j inputs in $\mathrm{Sub}(i)$ that $A^{(c,K)}$ queries, counting the queries of P^* and of the verifier. The only stage-h input in $\mathrm{Sub}(i)$ is i itself, so $N_{i,h} \in \{0, 1\}$.

Claim 1. If $D(i; w) = 1$, then $q_{i,c}(w) \le \sum_{j \in \mathrm{seg}(h)} \varepsilon_j \, \mathbb{E}_K[N_{i,j}(c, K)]$. Suppose $E_{h+1}(c, K)$ returns $y \in Y_{h+1}$ with candidate w and $u_h = i$. By invariant (2), the stage inputs $u_h, \ldots, u_{\bar{h}+1}$ of y are those of $A^{(c,K)}$, and their challenges are the table values $K(u_j)$. Condition (c) gives $D(u_{\bar{h}+1}; w) = 0$, while $D(u_h; w) = 1$. By condition (b) there is a stage $j \in \mathrm{seg}(h)$ on this path with $D(u_j; w) = 1$ and $K(u_j) \in \mathsf{Bad}_j(u_j; w)$. It remains to bound the probability of that event. Sample K lazily along the run of $A^{(c,K)}$. When A first queries a stage-j input $u \in \mathrm{Sub}(i)$, the input u, the bit $D(u; w)$, and the set $\mathsf{Bad}_j(u; w)$ are determined by the history, c, and the fixed w, while $K(u)$ is fresh. That query therefore lands in its bad set with probability at most ε_j. Every input on the output path is queried by A, so a union bound over the queried stage-j inputs in $\mathrm{Sub}(i)$ gives the claim. The bound holds for every fixed c and in every query order, because the whole table on $\mathrm{Sub}(i)$ is fresh. It covers $j = h$, where the one input is i, and it covers a later-stage input that embeds a guessed answer at i: such an input is counted like any other.

Claim 2. For every $j \in \mathrm{seg}(h)$,

$$\sum_i \mathbb{E}_J[N_{i,j}(J)] \le \mathbb{E}[Q_j] + 1 \qquad \text{and} \qquad \sum_i \Pr_{c_i}[P_{i,c_i} > 0] \le Q + 1.$$

Each sum ranges over the stage-h inputs i, and c_i is the restriction of a uniform table J to the complement of $\mathrm{Sub}(i)$. The sets $\mathrm{Sub}(i)$ of distinct stage-h inputs are disjoint. A stage-j input therefore lies in at most one of them, P^* queries at most Q_j stage-j inputs, and the verifier queries one. This is the first bound. For the second, 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-h input after the replacement only if $i = u_h(J)$. Hence $P_{i,c_i(J)} > 0$ for at most one input i beyond those whose set $\mathrm{Sub}(i)$ receives a query of P^*, and at most Q sets receive one. The second bound cannot be improved to $\mathbb{E}[Q_h] + 1$: a prover may query later-stage inputs in several sets $\mathrm{Sub}(i)$ without querying any stage-h input, and let their answers decide which i its proof uses.

Success. E_h returns $v = 1$ whenever step 1 does, except in the event Fail_h that step 1 returns y with $\mathsf{Good}_h(y) = 0$ and candidate w^0, and the loop ends with $y' \notin \mathcal{T}$ and $\mathsf{Cand}_h(y') = w^0$. Condition (a) gives $D(i; w^0) = 1$. For each stage-h input i let c_i be the restriction of the uniform table to the complement of $\mathrm{Sub}(i)$. The attempts of step 3 are independent copies of $E_{h+1}(c_i, K')$, independent of the first run given c_i, and the loop stops at the first with $v' = 1$ and input 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

$$\begin{aligned} \Pr[\mathsf{Fail}_h] &= \sum_i \mathbb{E}_{c_i}\Big[\sum_{w : D(i;w)=1} f_{i,c_i}(w) \frac{q_{i,c_i}(w)}{P_{i,c_i}} \Big] \\ &\le \sum_i \mathbb{E}_{c_i}\Big[\max_{w : D(i;w)=1} q_{i,c_i}(w) \Big], \end{aligned}$$

with no contribution when $P_{i,c_i} = 0$, since $\sum_w f_{i,c}(w) \le P_{i,c}$. These are joint probabilities for a fixed input i, with the event $u_h = i$ inside them; nothing is conditioned on the selected input. Each expectation is over the marginal law of its own c_i. By Claim 1 the maximum is at most $\sum_{j \in \mathrm{seg}(h)} \varepsilon_j \, \mathbb{E}_K[N_{i,j}(c_i, K)]$, and (c_i, K) is a uniform table. The first bound of Claim 2 now gives

$$s_h \ge s_{h+1} - \Pr[\mathsf{Fail}_h] \ge s_{h+1} - \sum_{j \in \mathrm{seg}(h)} (\mathbb{E}[Q_j] + 1) \, \varepsilon_j.$$

Cost. For fixed (i, c), step 3 is entered with selected input i with probability at most $P_{i,c}$ and makes $1/P_{i,c}$ attempts in expectation. Let $\Gamma_{i,c} := \mathbb{E}_K[\mathrm{cost}(E_{h+1}(c, K))\mathbf{1}[u_h(c, K) = i]]$, an unnormalised expectation. One attempt costs at most $1 + \Gamma_{i,c}$ in expectation, so by Wald's identity the expected cost of step 3 is at most $\sum_i \mathbb{E}_{c_i}[\mathbf{1}[P_{i,c_i} > 0](1 + \Gamma_{i,c_i})] \le (Q+1) + C_{h+1}$, using the second bound of Claim 2 and $\sum_i \mathbb{E}_{c_i}\Gamma_{i,c_i} = C_{h+1}$. Hence $C_h \le 2C_{h+1} + (Q+1)$.

Conclusion. Telescoping from $s_{m+1} = \varepsilon$ gives (191); the error of each plain stage appears once, because the stage lies in exactly one segment. 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 $\widehat{K} + (Q+1)(\widehat{K} - 1)$. For (192) use $\sum_{r \in H \cup N} Q_r \le Q$. In the interactive protocol there is one challenge node at each stage in place of an adversarially selected oracle input, and the same recurrences hold with $Q = 0$.

Remark 10.17 (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 the verifier requires to be zero. If these bits are not all zero, the verifier rejects every extension, so D can be kept equal to 1 and the answer need not belong to the bad set. A bad answer at a checkpoint or plain stage then has zero predicate bits and a bad challenge, and its probability is 2^{-g_r} times the probability that the challenge alone is bad, as in Proposition 10.3. For a challenge sampled from a fixed accepted family, ε_r is measured under that family's distribution.

Remark 10.18 (Structures selected by the statement). The stage types, arities, maps, and errors of a checkpoint structure may depend on the first message a_1, because every stage input encodes it. A node of the extractor reads them from the index tuple returned by its first run, and it keeps a later attempt only if that attempt has the same stage input. The proof of Theorem 10.16 then applies input by input. Let $\varepsilon^{(s)}$ be the sum of the stage errors and $\widehat{K}^{(s)}$ the tree size of the structure selected by $a_1 = s$. The knowledge error is at most $(Q+1)\max_s \varepsilon^{(s)}$ and the expected number of runs is at most $(Q+2)\max_s \widehat{K}^{(s)}$. Indeed, each recurrence charges a count of queried inputs. The inputs 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, and these quantities sum to at most $Q+1$ over s.

From the Shared Setup to Module-SIS The public key distribution is the stacked-prefix distribution of Section 4.3, not a product distribution over independent matrices. Let $\mathsf{Occ}(\mathsf{sch})$ contain every scheduled source-group occurrence at a nonterminal fold and the single terminal source occurrence. An occurrence u records its $\mathbf{A}$ matrix view and the bounds

$$K_{1,u},\ B_{\infty,u},\ K_{\mathsf{mul},2,u},\ B_{2,u},\ K_{2,u},\ B_{\mathsf{mul},2,u}$$

that its verifier checks for an extracted challenge difference and response difference. A Euclidean entry may be 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* $u^0_{S,h}$. If $(\mathbf{s}^0_{S,h,c,i})_{c,i}$ is the canonical algebraic decommitment fixed by preprocessing, this occurrence uses the weak certificate

$$\bar{c}^0_{S,h,c,i} = 1, \qquad \mathbf{r}^0_{S,h,c,i} = \mathbf{s}^0_{S,h,c,i}, \tag{193}$$

and records the schedule-certified envelopes

$$\begin{aligned} K_{1,u^0_{S,h}} &= 1, \\ B_{\infty,u^0_{S,h}} &\ge \max_{c,i} \|\mathbf{s}^0_{S,h,c,i}\|_{\infty,\mathrm{coef}}, \\ B_{2,u^0_{S,h}} &\ge \max_{c,i} \|\mathbf{s}^0_{S,h,c,i}\|_{2,\mathrm{coef}}. \end{aligned} \tag{194}$$

The last entry makes the first Euclidean pairing available when the protocol occurrence also 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.

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 schedule-defined equalities partition the occurrence positions into source-comparison classes. The commitment payload is not part of this partition: two extracted sources are compared only when their actual commitments are equal, and such occurrences necessarily lie in one class.

For u, u' in one class, define the coefficient envelope

$$\mathsf{E}_\infty(u,u') := K_{1,u'}B_{\infty,u} + K_{1,u}B_{\infty,u'}. \tag{195}$$

For the Euclidean route, let $\mathcal{P}(u,u') \subseteq \{1,2,3\}$ contain the pairings for which both occurrences certify the required bounds, and set

$$\begin{aligned}
\mathsf{E}_{2,1}(u,u') &:= K_{1,u'}B_{2,u} + K_{1,u}B_{2,u'},\\
\mathsf{E}_{2,2}(u,u') &:= K_{\mathsf{mul},2,u'}B_{2,u} + K_{\mathsf{mul},2,u}B_{2,u'},\\
\mathsf{E}_{2,3}(u,u') &:= K_{2,u'}B_{\mathsf{mul},2,u} + K_{2,u}B_{\mathsf{mul},2,u'}.
\end{aligned} \tag{196}$$

Each comparison class C selects one norm p_C. Its exact scheduled $\mathbf{A}$ collision bound is

$$\eta_{A,C} := \begin{cases} \max\limits_{u,u' \in C} \mathsf{E}_\infty(u,u'), & p_C = \infty,\\ \max\limits_{u,u' \in C} \min\limits_{r \in \mathcal{P}(u,u')} \mathsf{E}_{2,r}(u,u'), & p_C = 2. \end{cases} \tag{197}$$

The Euclidean choice is admissible only if $\mathcal{P}(u,u') \neq \varnothing$ for every ordered pair in C. In particular, the diagonal pair $u = u'$ prices two extractions from the same occurrence. For a $\mathbf{B}$ or $\mathbf{F}$ source-path view K shared by a class, let $\mathcal{D}_{K,u}$ be its verifier-certified input alphabet at protocol occurrence u, including all authenticated coordinates; at a canonical reference occurrence it is the canonical input alphabet fixed by the registered commitment profile. Define

$$\eta_{K,C} := \max_{u,u' \in C} \max_{x \in \mathcal{D}_{K,u},\, y \in \mathcal{D}_{K,u'}} \|x - y\|_{\infty,\mathrm{coef}}. \tag{198}$$

Thus a commitment scanned under two different accepted profiles is priced for their cross-profile difference, not only for either diagonal profile. Opening-side $\mathbf{D}/\mathbf{H}$ paths remain occurrence-local.

For an admissible schedule sch, let

$$\mathcal{I}(\mathsf{sch})$$

be the multiset of all Module-SIS instances induced by the schedule: for every source-comparison class, its $\mathbf{A}$ instance at (197) and its nonterminal $\mathbf{B}/\mathbf{F}$ source-path instances at (198); 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. The terminal class contributes only its $\mathbf{A}$ instance. Setup openings are ordinary source occurrences and hence enter the same comparison-class construction. Each element $I \in \mathcal{I}(\mathsf{sch})$ 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.

Lemma 10.19 (Shared-prefix breaks reduce to ordinary MSIS). *Let $\mathcal{B}$ be any PPT adversary that, given $(\mathsf{sch}, \mathbf{S})$ for a uniform flat setup vector $\mathbf{S} \leftarrow \mathbb{Z}_q^{N_{\mathsf{setup}}}$, outputs a matrix identifier $I \in \mathcal{I}(\mathsf{sch})$ and a short nonzero kernel vector for the corresponding matrix view. Define*

$$\mathsf{Adv}^{\mathsf{sch}}_{\mathsf{sp}}(\mathcal{B}) := \Pr\left[\begin{array}{c} \mathbf{S} \leftarrow \mathbb{Z}_q^{N_{\mathsf{setup}}};\ (I,\mathbf{x}) \leftarrow \mathcal{B}(\mathsf{sch},\mathbf{S}) \\ I \in \mathcal{I}(\mathsf{sch}) \ \wedge\ 0 \neq \mathbf{x} \in R_{q,d_I}^{m_I} \ \wedge\ \mathbf{M}_I(\mathbf{S})\mathbf{x} = 0 \ \wedge\ \|\mathbf{x}\|_{p_I,\mathrm{coef}} \le \eta_I \end{array}\right],$$

where $\mathbf{M}_I(\mathbf{S})$ is the matrix obtained from the same flat setup vector by its prefix view. Then there are ordinary Module-SIS adversaries $(\mathcal{B}_I)_{I \in \mathcal{I}(\mathsf{sch})}$ such that

$$\mathsf{Adv}^{\mathsf{sch}}_{\mathsf{sp}}(\mathcal{B}) \le \sum_{I \in \mathcal{I}(\mathsf{sch})} \mathsf{Adv}_{\mathsf{MSIS}^{(p_I)}_{q,d_I}(n_I,m_I,\eta_I)}(\mathcal{B}_I), \tag{199}$$

and each $\mathcal{B}_I$ has the running time of $\mathcal{B}$ plus polynomial overhead.

Table 7: The MSIS instances and their norm bounds for schedule sch. For $\mathbf{B}$ and $\mathbf{D}$, the collision bounds are the certified range-check alphabet bounds of Remark 10.24, with b the base of the level that range-checks the segment. For the $\mathbf{A}$ matrix, the schedule selects one norm specialization per source-comparison class. Its radius is the maximum mixed occurrence-pair envelope of (197); the usual coefficient and Euclidean formulas are its diagonal special cases. The $\mathbf{B}$ maps use the corresponding cross-profile alphabet diameter, while $\mathbf{D}$ maps remain occurrence-local. Every $\mathbf{F}/\mathbf{H}$ map has collision bound one, since its complete input is certified binary. A class containing a registered setup opening also contains its canonical reference profile from (194). Neither 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 (197)
$\mathbf{A}^{(\mathsf{term})}$	Terminal Inner Commitment	d_A	n_A	m_A	∞ or 2	(211) or (212)
$\mathbf{B}^{(C)}$	Source-Class Sliced Outer Commitment	d_B	n_B	m_B	∞	$\eta_{B,C}$
$\mathbf{D}^{(j)}$	Unsliced Opening Commitment	d_D	n_D	m_D	∞	$\eta_{D,j}$
$\mathbf{F}_\ell^{(C)}$	Source-Class Outer Chain Map $\ell \in \{1,\dots,L_F\}$	$d_{F,\ell}$	$n_{F,\ell}$	$m_{F,\ell}$	∞	1
$\mathbf{H}_\ell^{(j)}$	Opening-Local Chain Map $\ell \in \{1,\dots,L_H\}$	$d_{H,\ell}$	$n_{H,\ell}$	$m_{H,\ell}$	∞	1

For the sliced matrix $\mathbf{B}$, the length m_B is the width of the one matrix reused across all slices, not the total logical input length. A kernel vector of the complete stacked map restricts to a nonzero slice; zero-padding that restriction gives a kernel vector of the width-m_B matrix (Lemma 4.4). The matrix $\mathbf{D}$ is unsliced, so m_D is the full width of its ordered partial-evaluation input. For a source class C, $\eta_{B,C}$ is the maximum cross-profile diameter of (198). The occurrence-local quantity $\eta_{D,j}$ is the diameter of the alphabet certified at level j, namely $b_{\mathsf{range}} - 1$ for the balanced range. Every compression map has input alphabet $\{-1, 0\}$ under all scanning profiles, so $\eta_{F,C,\ell} = \eta_{H,j,\ell} = 1$. The fused binary constraint certifies their complete input spans (Remark 6.2).

Proof. Fix a matrix instance I and let the ordinary MSIS challenge be a uniform matrix $\mathbf{M}^\star \in R_{q,d_I}^{n_I \times m_I}$. The simulator samples a flat setup vector $\mathbf{S}$ subject only to the constraint $\mathbf{M}_I(\mathbf{S}) = \mathbf{M}^\star$, filling every coordinate not fixed by this view 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 (59)). Since $\mathbf{M}^\star$ is uniform, the resulting $\mathbf{S}$ is exactly uniform over $\mathbb{Z}_q^{N_{\mathsf{setup}}}$. The simulator then runs $\mathcal{B}(\mathsf{sch}, \mathbf{S})$. If $\mathcal{B}$ outputs the same matrix I together with a valid vector $\mathbf{x}$, the simulator returns $\mathbf{x}$ as a solution to the ordinary $\mathsf{MSIS}_{q,d_I}^{(p_I)}(n_I, m_I, \eta_I)$ challenge; otherwise it fails. The success probability of this solver is precisely the probability that $\mathcal{B}$ succeeds and names matrix I. Summing over all possible output matrices gives (199).

Corollary 10.20 (Shared-prefix reduction for an adaptive schedule). *Let $\mathcal{S}$ be a finite public family of admissible schedules, fixed independently of the setup, and let*

$$\mathbb{I}_{\mathcal{S}} := \bigcup_{\mathsf{sch} \in \mathcal{S}} \mathcal{I}(\mathsf{sch}).$$

Suppose an adversary, after seeing one uniform flat setup vector, chooses $\mathsf{sch} \in \mathcal{S}$ and outputs a valid short kernel vector for some $I \in \mathcal{I}(\mathsf{sch})$. There are ordinary Module-SIS adversaries $(\mathcal{B}_I)_{I \in \mathbb{I}_{\mathcal{S}}}$ such that the probability of this event is at most

$$\sum_{I \in \mathbb{I}_{\mathcal{S}}} \mathsf{Adv}_{\mathsf{MSIS}_{q,d_I}^{(p_I)}(n_I, m_I, \eta_I)}(\mathcal{B}_I). \tag{200}$$

The adversary may choose the schedule as a function of the setup; no conditioning on that choice is used.

Proof. For each $I \in \mathbb{I}_{\mathcal{S}}$, embed an ordinary Module-SIS challenge as the I-view of a uniform flat setup vector and sample the unused suffix uniformly, exactly as in Lemma 10.19. Run the original adversary, including its schedule selection, on this vector, and retain its output only when it selects a schedule containing I and names I. The simulated flat vector is unconditionally uniform. Summing over the matrix identifier output by the adversary gives (200).

Relaxed Source Extraction and Uniqueness Fix a nonterminal fold level h. Its opening method is part of the schedule: packing for $h \in \{0,1\}$ and evaluation trace for $h \geq 2$. For each source instance j, let S_j denote its folding ring and let

$$\iota_j : S_j \longrightarrow R_{q,d_{A,j}}$$

be the map through which a folding challenge acts on the ambient source. In coefficient packing, S_j is the challenge subring and ι_j is the embedding of (62); $\mathcal{L}_j$ and $\mathcal{V}_{j,c}$ are the packing partial and scalar-opening maps of Section 5.3. With evaluation trace, S_j is the full evaluation-trace folding ring, ι_j is the identity, $\mathcal{L}_j$ is the evaluation-trace partial map, and $\mathcal{V}_{j,c}$ is its public, tensor-reduction-scaled trace functional. Products of a challenge with an ambient source below mean multiplication by $\iota_j(c)$.

This distinction is internal to a fold. Both modes produce the same semantic successor type: an ordinary evaluation claim on a multilinear polynomial with base-field Boolean evaluations. An intermediate edge binds those evaluations through its ordinary outer commitment. The last nonterminal fold instead binds the canonical inner t state required by the terminal. Neither edge produces a tensor-reduction output claim. Whenever a later evaluation-trace receiver exists, that receiving fold performs the tensor reduction that converts its input. The terminal performs the same reduction on its input.

Definition 10.21 (Akita algebraic decommitment). *Fix one source commitment*

$$\mathsf{cm}_j := (\mathsf{shape}_{\mathsf{com},j}, \mathbf{u}_{\mathsf{pub},j}). \tag{201}$$

Its Src *component fixes the ordered source-vector layouts and decoder. The full shape fixes the block geometry, matrix views, slice maps, digit alphabets, and active compression path described in Section 5.1. An* Akita algebraic decommitment *of* cm_j *to the ordered decoded commitment-grid vectors* $(f^\star_{j,c})_c$ *is the commitment-side part of that construction. For every applicable* c *and* i*, its source blocks satisfy*

$$f^\star_{j,c} = \mathsf{Dec}_{\mathsf{shape}_{\mathsf{com},j}}((\mathbf{s}_{j,c,i})_{i\in[B_j]}), \qquad \mathbf{A}_j\mathbf{s}_{j,c,i} = \mathbf{G}_{b_{1,j},n_{A,j}}\hat{\mathbf{t}}_{j,c,i}.$$

The prescribed slice map places the inner-image digits $\hat{\mathbf{t}}_{j,c,i}$ *in the inputs of* (68)*, with* $P = n_{\mathsf{clm},j}$ *and* $B = B_j$*. The resulting* $\mathbf{B}_j$ *rows and active* $\mathbf{F}$ *chain satisfy* (71)*, or* (72) *at depth zero, and terminate at* $\mathbf{u}_{\mathsf{pub},j}$*. 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

$$\mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{com}}(\mathsf{pp}, \mathsf{cm}_j, (f^\star_{j,c})_c, \mathsf{st}^\star_{\mathsf{com},j}) = 1 \tag{202}$$

when these conditions hold. Its single-polynomial specialization instantiates $\mathsf{Rel}_{\mathsf{com}}$ *in Definition 3.7; the group form is the relation used by Akita's batch theorem. It contains the canonical state produced by the honest commitment algorithm. It requires neither the canonical gadget decomposition for* $\mathbf{s}_{j,c,i}$ *nor membership of the decoded vector in the image of an honest padding map. Partial opening evaluations, the* $\mathbf{D}/\mathbf{H}$ *path, the opening point, and the claimed value are excluded: they belong to one evaluation proof and may change when the same commitment is opened at another point.*

For an application batch, write its public statement as

$$\mathsf{x}_{\mathsf{batch}} := \left(\mathsf{sch}; \left(\mathsf{cm}_j, \mathbf{r}_j, (\pi_{j,c}, v_{j,c})_{c\in[n_{\mathsf{clm},j}]}\right)_{j\in[G]}\right). \tag{203}$$

An *Akita batch witness* consists of decoded full commitment-grid vectors $(f^\star_{j,c})_{j,c}$ and one algebraic decommitment state $\mathsf{st}^\star_{\mathsf{com},j}$ for every group occurrence. If one commitment occurs in several groups, the schedule-extraction theorem below establishes agreement of their vector families, unless it returns a matrix collision. Define

$$\begin{aligned}&\mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{batch}}(\mathsf{pp}, \mathsf{x}_{\mathsf{batch}}; (f^\star_{j,c})_{j,c}, (\mathsf{st}^\star_{\mathsf{com},j})_j) = 1\\ &\quad\iff \bigwedge_{j\in[G]} \mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{com}}(\mathsf{pp}, \mathsf{cm}_j, (f^\star_{j,c})_c, \mathsf{st}^\star_{\mathsf{com},j}) = 1 \ \wedge \bigwedge_{j,c} \widetilde{f^\star_{j,c}}(\mathbf{r}_j) = \phi_{j,c}v_{j,c}.\end{aligned} \tag{204}$$

The last equality is the padded claim verified by the protocol. If the statement additionally certifies $f^\star_{j,c} = \mathsf{Pad}_{j,c}(F_{j,c})$, then (75) recovers the natural-domain claim $\widetilde{F_{j,c}}(\pi_{j,c}(\mathbf{r}_j)) = v_{j,c}$. The Akita opening protocol does not include that padding-image predicate in $\mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{batch}}$.

The fixed folding statement. Fix one nonterminal fold with the G ordered source groups and public data of Sections 5.1 and 7.2. Thus the statement fixes $(\mathsf{cm}_j)_{j\in[G]}$, 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 (134).

Definition 10.22 (Weak opening for one fold). *A* weak opening *uses those layouts and commitment-opening rows with the following extraction relaxation. It contains* $(\mathbf{s}_{j,c,i}, \hat{\mathbf{t}}_{j,c,i}, \hat{\mathbf{e}}_{j,c,i}, \bar{c}_{j,c,i})$ *and all active compression-chain openings. For every* (j,c,i), $\bar{c}_{j,c,i}$ *is a unit in* S_j *and, for fixed challenge- and response-difference bounds, satisfies*

$$\|\iota_j(\bar{c}_{j,c,i})\|_1 \le \bar{\kappa}_{1,j}, \qquad \|\iota_j(\bar{c}_{j,c,i})\mathbf{s}_{j,c,i}\|_{\infty,\mathrm{coef}} \le \bar{\beta}_{\infty,j}.$$

On a Euclidean route it also satisfies

$$\|\iota_j(\bar{c}_{j,c,i})\|_{\mathsf{mul},2} \le \bar{\kappa}_{\mathsf{mul},2,j}, \qquad \|\iota_j(\bar{c}_{j,c,i})\mathbf{s}_{j,c,i}\|_{2,\mathrm{coef}} \le \bar{\beta}_{2,j}.$$

The source blocks, inner-image digits, and $\mathbf{B}/\mathbf{F}$ *openings satisfy Definition 10.21. 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* (139) *and the active compression rows or the depth-zero row* (72). *Recompose the partial digits in base* $b_{1,j}$ *and apply the selected mode's fixed decoding from base-field coordinates to recover* $e_{j,c,i}$ *(Equations* (85) *and* (93)*). The remaining rows enforce*

$$e_{j,c,i} = \mathcal{L}_j(\mathbf{G}_{b_{j,M_j}}\mathbf{s}_{j,c,i}), \qquad v_{\mathsf{fold}} = \sum_{j\in[G]}\sum_{c\in[n_{\mathsf{clm},j}]} \vartheta_{j,c}\,\mathcal{V}_{j,c}((e_{j,c,i})_{i\in[B_j]}).$$

Here v_{fold} *is* v_{root} *under coefficient packing and* $\bar{v}_{\mathsf{root}}$ *under evaluation trace. Every digit vector lies in the alphabet certified for its segment. This is the ordinary core relation when* $G = 1$.

The extracted source. For a weak opening ω, let $\mathsf{Src}(\omega)$ be its *source projection*: exactly the source blocks, inner-image digits, and $\mathbf{B}/\mathbf{F}$ chain openings. It discards every challenge difference, partial opening evaluation and its digits, $\mathbf{D}/\mathbf{H}$ opening, evaluation point, and claimed value. It is therefore an Akita algebraic decommitment in the sense of Definition 10.21. The additional bounded products $\iota_j(\bar{c}_{j,c,i})\mathbf{s}_{j,c,i}$ are the weak certificates used to prove that this source projection is extraction-unique.

Lemma 10.23 (Weak binding for one fold). *Consider two weak openings, in the sense of Definition 10.22, for the same public fold instance and payloads. If they differ on any semantic block witness* $\mathbf{s}_{j,c,i}$ *or carried digit vector, then one can deterministically output a short nonzero kernel vector for* $\mathbf{A}_j$, $\mathbf{B}_j$, $\mathbf{D}$, *or one of their compression maps. An* $\mathbf{A}_j$ *collision has coefficient norm at most* $2\bar{\kappa}_{1,j}\bar{\beta}_{\infty,j}$*; every other collision is bounded by the diameter of the corresponding verifier-certified alphabet.*

More precisely, for a coordinate on which the openings differ, set $\mathbf{r}_i := \iota_j(\bar{c}_i)\mathbf{s}_i$ *and* $\mathbf{r}'_i := \iota_j(\bar{c}'_i)\mathbf{s}'_i$. *In the bounds below, each norm of* $\bar{c}_i$ *means the corresponding norm after applying* ι_j. *The same kernel vector satisfies all of the following bounds:*

$$\|\mathbf{z}_A\|_{\infty,\mathrm{coef}} \le \|\bar{c}'_i\|_1\|\mathbf{r}_i\|_\infty + \|\bar{c}_i\|_1\|\mathbf{r}'_i\|_\infty, \tag{205}$$

$$\|\mathbf{z}_A\|_{2,\mathrm{coef}} \le \min\left\{\begin{array}{l}\|\bar{c}'_i\|_1\|\mathbf{r}_i\|_2 + \|\bar{c}_i\|_1\|\mathbf{r}'_i\|_2,\\ \|\bar{c}'_i\|_{\mathsf{mul},2}\|\mathbf{r}_i\|_2 + \|\bar{c}_i\|_{\mathsf{mul},2}\|\mathbf{r}'_i\|_2,\\ \|\bar{c}'_i\|_2\|\mathbf{r}_i\|_{\mathsf{mul},2} + \|\bar{c}_i\|_2\|\mathbf{r}'_i\|_{\mathsf{mul},2}\end{array}\right\}. \tag{206}$$

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*

$$\|\mathbf{z}_A\|_{\infty,\mathrm{coef}} \le 2\bar{\kappa}_1\bar{\beta}_\infty, \tag{207}$$

$$\|\mathbf{z}_A\|_{2,\mathrm{coef}} \le 2\min\left\{\bar{\kappa}_1\bar{\beta}_2, \bar{\kappa}_{\mathsf{mul},2}\bar{\beta}_2, \bar{\kappa}_2\bar{\beta}_{\mathsf{mul},2}\right\}. \tag{208}$$

If the first and second openings instead have the occurrence profiles u *and* u', *respectively, the right-hand side of* (205) *is at most* $\mathsf{E}_\infty(u,u')$. *Each available line of* (206) *is at most the corresponding* $\mathsf{E}_{2,r}(u,u')$ *from* (196).

Remark 10.24 (Certified versus honest digit alphabets). The collision norms above are priced at the alphabet the protocol *enforces*, not the alphabet the honest prover uses. The carried source digit vectors $\hat{\mathbf{t}}, \hat{\mathbf{e}}$ are honestly balanced base-b_1 gadget digits. For a cheating prover, however, shortness is enforced 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 shared balanced alphabet $\{-b^\star/2, \dots, b^\star/2-1\}$ of that level's certified range base $b^\star$. Completeness therefore requires $b_1 \le b^\star$, which admissibility enforces (Section 9.3), and extraction may assume only the certified bound: the $\mathbf{B}$ and $\mathbf{D}$ collision norms are $b^\star - 1$, and the schedule inventory is priced there ($b_1 - 1$ would *under*price the adversary whenever $b_1 < b^\star$). Every compression map instead has collision norm one: its complete digit span participates in the fused constraint $w(w+1) = 0$, which enforces $\{-1, 0\}$ (Remark 6.2). The folded response obeys the same rule: its planes are bounded not by whatever norm the honest prover grinds them to, but by (109), the recomposition of the alphabet that the scanning level certifies. Security sizing and support derivation therefore use the same authenticated alphabet for each map, and a collision norm is a property of the level that scans a segment rather than 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 $\mathbf{B}/\mathbf{F}$ comparison-class instance uses the maximum cross-profile diameter defined in (198). For $\mathbf{F}$ this diameter remains one across profiles. The opening-local $\mathbf{D}$ path uses its scanning level's own diameter, while $\mathbf{H}$ again has diameter one.

Proof (Proof of Lemma 10.23). Apply Lemma 4.4 to every $\mathbf{B}_j/(\mathbf{F}_{h,j})_h$ 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.

Assume therefore that all carried digit vectors agree, in particular $\hat{\mathbf{t}}_{j,c,i} = \hat{\mathbf{t}}'_{j,c,i}$ and $\hat{\mathbf{e}}_{j,c,i} = \hat{\mathbf{e}}'_{j,c,i}$ for every coordinate. If the two weak openings still differ, choose (j, c, i) with $\mathbf{s}_{j,c,i} \ne \mathbf{s}'_{j,c,i}$ and omit these indices below. The inner consistency equations give $\mathbf{A}_j \mathbf{s}_i = \mathbf{G}_{b_{1,j}, n_{A,j}} \hat{\mathbf{t}}_i = \mathbf{A}_j \mathbf{s}'_i$. Define

$$\mathbf{z}_A := \iota_j(\bar{c}'_i)\big(\iota_j(\bar{c}_i)\mathbf{s}_i\big) - \iota_j(\bar{c}_i)\big(\iota_j(\bar{c}'_i)\mathbf{s}'_i\big).$$

The two openings are compared across branches of the Fiat–Shamir tree that draw their folding challenges from different transcript prefixes, so $\bar{c}_i$ and $\bar{c}'_i$ need not agree and both denominators must be cleared. This produces the two summands in every norm bound below. Since $\iota_j(\bar{c}_i)$ and $\iota_j(\bar{c}'_i)$ are units and $\mathbf{s}_i \ne \mathbf{s}'_i$, the vector $\mathbf{z}_A$ is nonzero. Moreover $\mathbf{A}_j \mathbf{z}_A = 0$.

Apply (43) separately to the two summands to obtain (205). Applying each of the three Euclidean inequalities in (44), again term by term, gives the three lines inside (206). Taking the uniform bounds for the two weak openings yields (207) and (208). This proves every stated specialization for the same nonzero $\mathbf{A}$-kernel vector.

Corollary 10.25 (Source-extractor uniqueness across opening profiles). *Fix a schedule. For a source-group occurrence $u = (h, j)$, let*

$$\varphi(u) := \mathsf{cm}_{h,j} = (\mathsf{shape}_{\mathsf{com},j}, \mathbf{u}_{\mathsf{pub},j}), \qquad \varphi_h := (\varphi(h, j))_{j \in [G_h]} \tag{209}$$

be its source commitment and the ordered projection of its fold input, respectively. Write $\mathsf{Src}_u(\omega)$ for the component of $\mathsf{Src}(\omega)$ belonging to occurrence u. Consider extracted weak openings ω, ω' containing occurrences u, u', possibly at different points and with different claimed values and opening-side $\mathbf{D}/\mathbf{H}$ payloads, for which u, u' lie in the same source-comparison class and $\varphi(u) = \varphi(u')$. If $\mathsf{Src}_u(\omega) \ne \mathsf{Src}_{u'}(\omega')$, 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 norm p_C at most $\eta_{A,C}$ for their comparison class C; a source-path collision is bounded by that class's cross-profile certified-alphabet diameter. 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, with $\varphi(u^0_{S,h})$ equal to its registered commitment. On that side use the canonical algebraic decommitment and the certificate (193); no evaluation point or opening-side data are needed. Its collision bounds are the mixed envelopes with the reference profile (194).

Proof. Apply Lemma 4.4 to the two openings of the common $\mathbf{B}/\mathbf{F}$ payload, using its scheduled depth. On the raw route, a disagreement gives a $\mathbf{B}$ collision directly. On the compressed route, a difference in its chain digits gives a $\mathbf{B}$ or $\mathbf{F}$ collision, whose norm is at most the cross-profile alphabet diameter recorded for the

comparison class. Otherwise the inner-image digits $\hat{\mathbf{t}}$ agree. If a source block still differs, use its two weak folding factors to clear denominators exactly as in the proof of Lemma 10.23. Equations (195)–(197) give its scheduled norm bound. The opening point and the $\mathbf{D}/\mathbf{H}$ path do not enter either argument.

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 (194) supplies the second occurrence profile in the same mixed-envelope calculation.

Incoming tensor-reduction data. Let h be an evaluation-trace receiver. It instantiates the ordered multi-instance interface of Section 5.1 and the shared tensor reduction of Section 5.2. A nonterminal receiver has $h \geq 2$; 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. Denote claim c's Boolean evaluation vector, native point, and value by $f_{h,g,c}$, $\mathbf{r}_{h,g}$, and $v_{h,g,c}$, so its input assertion is $\widetilde{f_{h,g,c}}(\mathbf{r}_{h,g}) = v_{h,g,c}$. At a nonterminal receiver, $P_{h,g}$ is its ordinary outer commitment; at the terminal it is the predecessor-bound canonical inner t state. The terminal has exactly one recursive-witness group and no pending setup opening.

Definition 10.26 (Incoming claims for tensor reduction). *For the incoming data above, the receiver uses the following boundary rules. The schedule fixes every group descriptor, basis, packing bijection* $\mathsf{Pack}_{h,g}$*, and tail arity* $m_{h,g}$*; in particular,*

$$g_{h,g,c} := \mathsf{Pack}_{h,g}(f_{h,g,c})$$

is the packed multilinear polynomial 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. The transcript binds these descriptors together with the incoming payloads, native points, and values before the receiver begins its shared tensor-reduction prefix. Set*

$$m_{\max,h} := \max_g m_{h,g}, \qquad N_{\mathsf{clm},h} := \sum_g |I_{h,g}|.$$

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*

$$\theta_{h,g} = A_{\eta_h,g}(\boldsymbol{\rho}_{h,g}) \prod_{t=m_{h,g}}^{m_{\max,h}-1} (1-\rho_{h,t}), \qquad h_{h,g,c} = \theta_{h,g} g_{h,g,c}(\boldsymbol{\rho}_{h,g}).$$

The verifier rejects if any $\theta_{h,g}$ *is zero. On the identity route, the prefix and its challenges are omitted,* $\theta_{h,g} = 1$*, and the canonical packing identification forwards each claim. Each group's input point remains part of its bound statement.*

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 (80); its one-shot bound replaces $N_{\mathsf{clm},h} - 1$ by $\mathbf{1}_{\{N_{\mathsf{clm},h}>1\}}$, and its claim-batching tree factor is $N_{\mathsf{clm},h}$. The subsequent post-payload claim batch has coordinate-wise charge

$$\varepsilon_{\mathsf{grp},h} := \frac{N_{\mathsf{clm},h}-1}{|E_h|} \tag{210}$$

and tree factor $N_{\mathsf{clm},h}$; its one-shot failure is at most $\mathbf{1}_{\{N_{\mathsf{clm},h}>1\}}/|E_h|$.

Lemma 10.27 (Tensor reduction in the receiving fold). *Fix a boundary in the sense of Definition 10.26. Suppose the extractor has 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. Below every leaf of this prefix, suppose there is a descendant-certified evaluation-trace source. Each source's canonical base-coordinate unpacking must authenticate each fixed payload $P_{h,g}$ under the scheduled source relation and satisfy 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 in Definition 10.26. The identity route has no row-batching, tensor claim-batching, or sum-check challenges and contributes zero tensor-reduction error.

In particular, the first evaluation-trace receiver composes the predecessor extractor with its own tensor-reduction prefix. A nonterminal receiver then runs its fold and either produces an ordinary opening or binds the predecessor state for the terminal. When setup offloading supplies recursive-witness and setup groups, they enter the same shared tensor reduction. The incoming claims remain separately bound; each output claim uses its prescribed prefix of the new sum-check point.

Proof. 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. Finally, Lemma 10.13 applied to the nested binary η tree forces every tensor-row discrepancy to vanish identically. 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.

After the shared opening payload is bound, the fresh evaluation-trace claim batch separates the final products with the charge in (210). Cylindrical extension changes only the sum-check domain; 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.

Terminal statement and transcript order. Every admissible schedule ends in the evaluation-trace terminal of Section 8.2. It 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. 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. We analyze the terminal that sends every block; Lemma 10.45 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 receiver-owned tensor reduction and direct terminal checks follow Section 8.2, including its transcript order. In particular, 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. The terminal then fixes the clear partials $(e_i)_i$ before sampling $(c_i)_i$, absorbs $\mathbf{z} = \sum_i c_i \mathbf{s}_i$, and verifies (161)–(163) 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 raw terminal challenge ℓ_1 norm and let $Z_{\infty,\mathsf{term}}$ bound the accepted clear response coefficient norm. The terminal source occurrence therefore has

$$K_{1,\mathsf{term}} = 2\kappa_{1,\mathsf{term}}, \qquad B_{\infty,\mathsf{term}} = 2Z_{\infty,\mathsf{term}}.$$

Its diagonal comparison-class radius on the coefficient route is

$$\eta_{A,\mathsf{term},\infty} = 8\kappa_{1,\mathsf{term}} Z_{\infty,\mathsf{term}}. \tag{211}$$

On a Euclidean route, let Γ_{term} be the enforced raw challenge operator-norm bound and let $S_{\max,\mathsf{term}}$ be the enforced response squared-norm bound. Then

$$K_{\mathsf{mul},2,\mathsf{term}} = 2\Gamma_{\mathsf{term}}, \quad B_{2,\mathsf{term}} = 2\sqrt{S_{\max,\mathsf{term}}}, \quad \eta^2_{A,\mathsf{term},2} = 64\Gamma^2_{\mathsf{term}} S_{\max,\mathsf{term}}. \tag{212}$$

The terminal **A** rank is admissible only if it meets the security floor for the selected one of these two instances.

Proposition 10.28 (Akita terminal extraction certificate). *For the terminal and bounds specified above, the following holds. 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* **A***-matrix collision. The exact terminal equations add no field error. Its fold challenges contribute*

$$\varepsilon_{\mathsf{term}} := \frac{B_{\mathsf{term}}}{|\mathcal{C}^{\mathsf{acc}}_{\mathsf{term}}|}, \tag{213}$$

while the terminal tensor-reduction term is charged separately. There is no terminal claim-batching term because the terminal has one claim.

Proof. 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, choose two children with challenges $\mathbf{c}^{(i,0)}$ and $\mathbf{c}^{(i,1)}$ that agree outside coordinate i and differ at i, as guaranteed by the CWSS structure. Let their responses be $\mathbf{z}^{(i,0)}$ and $\mathbf{z}^{(i,1)}$, and 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 **A** equations and cancelling Δc_i gives $\mathbf{A}_{\mathsf{term}}\mathbf{s}_i^\star = \mathbf{t}_i$. 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, they both map to the same $\mathbf{t}_i$. Clearing their unit challenge differences as in Lemma 10.23 gives the short **A** collision bounded by (211) or (212).

The trace equation has no fold challenge. The verifier checks it on any accepting child against the fixed pair $(\theta_{\mathsf{term}}, v_{\mathsf{tensor}})$; it is not obtained by fold-challenge subtraction. It connects the extracted partials to the tensor reduction output, and Lemma 10.27 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 (213); every remaining terminal check is an exact equality.

Corollary 10.29 (Radius-to-collision bookkeeping). *Suppose each raw folding challenge c and each accepted folded response* $\mathbf{z}$ *satisfy*

$$\|c\|_1 \le \kappa_1, \qquad \|c\|_{\mathsf{mul},2} \le \Gamma, \qquad \|c\|_2 \le \kappa_2,$$

and

$$\|\mathbf{z}\|_2 \le Z_2, \qquad \|\mathbf{z}\|_{\mathsf{mul},2} \le Z_{\mathsf{mul}}.$$

Then the Euclidean **A***-collision extracted from two weak openings obeys*

$$\|\mathbf{z}_A\|_{2,\mathrm{coef}}^2 \le 64 \min\left\{\kappa_1^2 Z_2^2, \Gamma^2 Z_2^2, \kappa_2^2 Z_{\mathsf{mul}}^2\right\}. \tag{214}$$

Proof. Take centered integer lifts of the raw challenges and accepted responses, but form their differences over $\mathbb{Z}$, without reducing those differences 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 now reduce the final collision coefficient-wise to its centered representative modulo q; this cannot increase its Euclidean norm. Squaring gives (214). This argument does not assume that coefficient-wise reduction contracts a convolution operator norm, and requires no no-wrap premise.

The quantities Z_2 and Z_{mul} in Corollary 10.29 are measured on centered ring coefficients. If a protocol instead certifies a norm before applying the extension-field packing map ψ, it first applies the logical-to-ring conversion factor of Section 3.1, exactly once. It does not apply that factor to an already packed $\mathbf{z}$ or to a $\mathbf{z}$ reconstructed from base-field digit planes.

Definition 10.30 (Nested coordinate-wise fold family). *Fix a nonterminal fold with G source groups and let*

$$\mathcal{Q} := \{(j, c, i) : j \in [G],\ c \in [n_{\mathsf{clm},j}],\ i \in [B_j]\}$$

index its folding challenges. A nested coordinate-wise fold family contains one transcript prefix ending immediately before this fold challenge and $|\mathcal{Q}| + 1$ folding-challenge children forming a coordinate-wise 2-special-sound structure. For every $q \in \mathcal{Q}$, choose a pair $\mathbf{c}^{(q,0)}, \mathbf{c}^{(q,1)}$ from these children that agree outside coordinate q and differ at q. Their difference at q is a unit with the norm prescribed by its source instance. The pairs need not share a common child.

Below every folding-challenge child, the family contains a complete accepting mixed subtree, as specified by Lemma 10.14, for the public-coin rounds that follow that challenge in the protocol. These include 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.

Call the family descendant-certified *if the recursive extractor has already authenticated every outgoing opening claim at the leaves of these strong-check subtrees. For one fixed folding child, all such openings under the same outgoing commitment must yield the same commitment-side source vector. Otherwise Corollary 10.25 already returns a matrix collision. At the tail, Proposition 10.28 supplies the descendant certificate directly.*

Theorem 10.31 (One-fold backward extraction). *Fix one nonterminal fold of an admissible schedule. Suppose its outgoing opening claims have already been authenticated by the recursive suffix, so that the extractor is given a descendant-certified nested coordinate-wise fold family in the sense of Definition 10.30. Then 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 j, the extracted coefficient-difference bound is $\bar{\beta}_{\infty,j} = \Delta^{\mathsf{cert}}_{f,j}$. If the schedule enforces a Euclidean radius profile from Corollary 10.29, 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. Taking $G = 1$ recovers the single-group fold. 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. The extractor proceeds in three steps. It first recovers the exact branch relation below each folding-challenge child, then compares the authenticated digit paths across children, and finally subtracts coordinate siblings.

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. The scheduled quotient-lift or quotient-free extractor then recovers the original mixed-dimension ring identities. Thus this child yields one exact instance of every row in 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. It also yields the certified digit and response bounds. Repeat this procedure independently below every folding-challenge child. Later challenges disappear at this point because every recovered identity holds over its native ring.

Compare every commitment path and the shared opening path across these exact branch relations, using the scheduled depth in Lemma 4.4. A disagreement on a raw path gives a $\mathbf{B}$ or $\mathbf{D}$ collision directly; a disagreement on a compressed path gives a collision in its source matrix or one of its chain matrices. Assume from now on that the corresponding source and chain digits agree across all branches.

Fix $q = (j, c, i) \in \mathcal{Q}$ and subtract the exact relation for child $(q, 0)$ from that for child $(q, 1)$. Every other folding term cancels inside instance j. With $\bar{c}_{j,c,i} := c^{(q,1)}_{j,c,i} - c^{(q,0)}_{j,c,i}$, the response rows give

$$\iota_j(\bar{c}_{j,c,i})\mathbf{s}_{j,c,i} = \mathbf{z}^{(q,1)}_j - \mathbf{z}^{(q,0)}_j.$$

The certified response digits bound this difference by $\Delta^{\mathsf{cert}}_{f,j}$. Subtracting the two fold-consistency and fold-evaluation rows, then cancelling the challenge difference in the ring of each row, gives

$$\mathbf{A}_j \mathbf{s}_{j,c,i} = \mathbf{G}_{b_{1,j},n_{A,j}} \hat{\mathbf{t}}_{j,c,i}, \qquad e_{j,c,i} = \mathcal{L}_j(\mathbf{G}_{b_j,M_j} \mathbf{s}_{j,c,i}).$$

The first inversion is in the ambient ring through $\iota_j(\bar{c}_{j,c,i})$, while the second is in the scheduled opening ring S_j (and its scalar extension when needed). With coefficient packing, Lemma 4.3 supplies both units from the same carrier difference. With evaluation trace, S_j is the ambient folding ring and ι_j is the identity, so the schedule's unit-difference condition supplies both inversions directly. The resulting difference also satisfies any enforced Euclidean bounds. Repeating this argument for every q extracts all semantic blocks. The common digits already satisfy their compression chains and scheduled public opening row because each strongly extracted branch satisfies every row exactly. They therefore form the claimed weak opening. At no point did the proof compare the later challenge values of two folding children. The final uniqueness statement is Corollary 10.25 applied to the two extracted weak openings.

The theorem is local to one fold. Its descendant certificate is supplied by the terminal at the last nonterminal fold and then by backward induction at each preceding fold. It therefore states the extraction step used in the schedule proof, rather than a standalone knowledge-soundness claim for an unauthenticated outgoing opening.

The exact branch relation. The first step of the proof above is the only one that reads the subtree below a folding-challenge child. We isolate what it delivers. Fix a nonterminal fold and a transcript prefix u through the messages of steps 3 and 4 of Figure 5. Thus u contains the fold challenges, the successor payload, and any clear norm claims. A table $\mathbf{w}$ on $\{0,1\}^{\mu'}$, read in the witness order of (106), *satisfies the exact branch relation of* u 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 6) holds as an identity in its native ring, for the commitments, opening payload, scalar row, and fold challenges in u; and
(iii) on a Euclidean route, the norm claims in u are true for the response digits in $\mathbf{w}$: the decoded integer $\mathcal{E}_{\mathsf{resp}}$ equals $\mathcal{E}_{\mathsf{int}}(\hat{\mathbf{z}})$, and on the digit-expanded route every claim $p_{t,h,k}$ is the residue of the integer inner product that it names.

Three features of this relation are used below. First, it can be decided in polynomial time from u and $\mathbf{w}$, so an extractor can test it. Second, it refers to no later challenge and to no short opening of the successor payload, so it is a property of the pair $(u, \mathbf{w})$ alone. Conditions (ii) and (iii) do not even depend on the quotient entries of $\mathbf{w}$, which enter only through their alphabet in (i). Third, it does not include the verifier's own checks on u, namely canonical decoding of the norm claims, the integer reconstruction (120), and the comparison $\mathcal{E}_{\mathsf{resp}} \le S_{\max}$; these are required separately. When they pass and (iii) holds, the integer response norm is at most $S_{\max}$.

Corollary 10.32 (One-fold extraction from exact branch relations). *Fix one nonterminal fold of an admissible schedule and a transcript prefix ending immediately before its fold challenge. Consider* $|\mathcal{Q}| + 1$ *extensions of this prefix through steps 3 and 4 whose fold challenges form the coordinate-wise structure of Definition 10.30. Suppose that each extension passes every verifier check that it determines, and that each 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. The second and third steps of the proof of Theorem 10.31 use 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) and the passing comparison with $S_{\max}$ supply the response bound. The two steps use nothing else.

Strong Checks and Error Accounting

Lemma 10.33 (Per-claim soundness of public batching). *Consider the coefficient-packing application root prescribed by (160). Fix the source commitments, opening payload, points, and claimed values, together with the uniquely bound partial evaluations, before the verifier draws ζ_{batch}. Let $\bar{v}_\ell$ be the padded value extracted for global claim ℓ. If $\bar{v}_\ell \neq \phi_{j(\ell),c(\ell)} v_{j(\ell),c(\ell)}$ for some $\ell \in [\Lambda]$, then*

$$\sum_{\ell\in[\Lambda]} \zeta_{\mathsf{batch}}^{\ell} \bar{v}_\ell = \sum_{\ell\in[\Lambda]} \zeta_{\mathsf{batch}}^{\ell} \phi_{j(\ell),c(\ell)} v_{j(\ell),c(\ell)}$$

holds with probability at most $(\Lambda-1)/|E|$.

Proof. The difference is a nonzero polynomial in ζ_{batch} of degree at most $\Lambda-1$. The claim follows from 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 let δ_f be 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*

$$\mathcal{E}_{\mathsf{int}}(\hat{\mathbf{z}}) = \mathcal{E}_{\mathsf{resp}} \le S_{\max}, \qquad \|\mathbf{z}\|_{2,\mathrm{coef}}^2 \le \mathcal{E}_{\mathsf{resp}}. \tag{215}$$

The additional random-linear-combination error is at most

$$\varepsilon_{\mathsf{phys}}^{\mathsf{rlc}} = \begin{cases} 2|E|^{-1}, & \text{direct norm check,} \\ (N_{\mathsf{pair}}+\delta_f)|E|^{-1}, & \text{digit-expanded norm reconstruction.} \end{cases} \tag{216}$$

At the Akita terminal, the verifier obtains (215) by decoding the response and computing the integer square sum directly. The terminal check has no corresponding sum-check or random-linear-combination error.

Proof. For the direct route, Lemma 6.3 gives the integer equality. For the digit-expanded route, it follows from Lemma 6.4 and Proposition 6.5. Injectivity of the response-digit map and monotonicity under centered reduction give the second inequality in (215).

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 degree at most δ_f on the digit-expanded route, contributing at most $1/|E|$ and $\delta_f/|E|$, respectively. The digit-expanded route also batches the N_{pair} inner-product claims, adding $(N_{\mathsf{pair}}-1)/|E|$. Summing gives (216). The terminal performs the same integer calculation directly on the revealed coefficients.

Corollary 10.35 (Route-specific inner-commitment collision). *Fix one source instance and its scheduled $\mathbf{A}$ matrix. Suppose every raw challenge satisfies $\|c\|_1 \le \kappa_1$, and every accepted response on the coefficient route satisfies $\|\mathbf{z}\|_{\infty,\mathrm{coef}} \le Z_\infty$. The extracted coefficient-norm collision then has radius*

$$p_A = \infty, \qquad \eta_{A,\infty} = 8\kappa_1 Z_\infty.$$

If the route is Euclidean, suppose instead that every accepted challenge has multiplication-operator norm at most Γ and that the recursive check or terminal enforces the bound $S_{\max}$. Suppose also that the centered integer lifts of differences of accepted challenges do not wrap coefficient-wise modulo q, so the operator-norm triangle inequality applies. The extracted collision then satisfies

$$p_A = 2, \qquad \eta_{A,2}^2 = 64\Gamma^2 S_{\max}.$$

For a coefficient-carrier source, the certified carrier bound is also the operator bound of the embedded challenge acting on the ambient response by Lemma 4.3. No lane or extension-degree factor is added. The planner may select this Euclidean radius only when the schedule 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. The schedule may use either exact radius or an upward-rounded radius in its Module-SIS inventory.

Proof. Taking two accepting openings doubles the raw challenge and response radii. The two terms that clear their weak-opening denominators supply the remaining factor of two. The coefficient statement follows from (207). The Euclidean statement follows from Lemma 10.34 and Corollary 10.29. An upward-rounded radius still contains every collision produced by the reduction.

For source instance j, a challenge family drawn from an ℓ_1-ball of mass ω_j has $\bar{\kappa}_{1,j} \le 2\omega_j$. The coefficient-norm $\mathbf{A}$ instance of that source is therefore priced at $\eta_{A,j} = 4\omega_j \Delta^{\mathsf{cert}}_{f,j}$.

The Euclidean alternative is the second route in Corollary 10.35. If the schedule certifies an exact operator bound on challenge differences, it may use that bound in place of 2Γ before applying (208); the stated route uses separate radius bounds on the two raw challenges.

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 ξ_{ring}-special sound, where*

$$\xi_{\mathsf{ring}} := \begin{cases} 2d_{\max}, & \mathsf{RingCheck} = \mathsf{QuotientLift}, \\ d_{\max}, & \mathsf{RingCheck} = \mathsf{QuotientFree}. \end{cases} \tag{217}$$

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 stated special-soundness 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, using Lemma 10.13; coordinate pairs alone would not suffice. The final claim follows from the degree-3 summand in (158) and Theorem 3.5.

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 $d_1, \dots, d_S$ and interstage frontiers of sizes $m_1, \dots, m_{S-1}$, then the digit range check contribution to the interactive knowledge error is*

$$\varepsilon_{\mathsf{tree}} = \sum_{j=1}^{S} \frac{\mu'(d_j + 1)}{|\mathbb{F}_{q^k}|} + \sum_{j=1}^{S-1} \frac{m_j - 1}{|\mathbb{F}_{q^k}|}, \tag{218}$$

where μ' is the extended-witness variable count. The range point τ_0 additionally uses a nested binary tree of depth μ': once its descendants certify the sum in (112), Lemma 10.13 forces every Boolean range residual to be zero. Its charge $\mu'/|E|$ is listed separately in (220).

The binary-support residual is multilinear in $\mathbf{r}_{\mathsf{virt}}$. This point is already supplied by the last range sum-check's nested scalar challenge tree. After its descendants certify the fused relation and separate ζ_{bin}, two children per coordinate of that tree suffice for binary-support interpolation. It adds no new tree factor.

Remark 10.38 (Opening-row binding). The extracted partials satisfy the field-level opening row already included in the relation: (133) with coefficient packing, or (134) with evaluation trace. The same τ_1 batches this row with the commitment, fold, and compression rows. It therefore contributes no separate term to (220); its row-batching and relation-check errors are already charged there. The preceding tensor reduction remains a separate strong reduction charged by (222).

The nested strong–weak invariant. SuperNeo formalizes two interfaces for precisely this lattice-extraction problem [83, Defs. 9–10, Thm. 8]. A weak interactive reduction extracts a witness for a relaxed relation and guarantees that 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, but not that sequential theorem as a black box. In protocol order, a folding challenge precedes the ring-reduction, range, norm, and relation challenges, and its outgoing claim is authenticated only by the recursive suffix. Moreover, the outgoing commitment is formed after the fold challenge and may differ across folding children, so it does not provide the common intermediate projection required at that cut by SuperNeo's strong-reduction definition. Thus Akita's weak fold and strong checks are

nested rather than arranged as a strong reduction followed by a weak reduction. The corresponding relaxed relation is Definition 10.22; the public projection is φ_h from (209); and the witness subject to uniqueness is $\mathsf{Src}(\omega)$, not the complete weak opening. This last distinction is necessary because partial opening evaluations and $\mathbf{D}/\mathbf{H}$ paths may legitimately change with the opening point.

The composition can be carried out in two ways. Given a complete accepting transcript tree, the recursive extractor first authenticates the outgoing source under one folding child. The later range, norm, ring-reduction, and sum-check extractors then recover the exact native-ring branch relation by interpolation. If two recursive extractions under the same outgoing commitment disagree on their source projection, Corollary 10.25 already gives a matrix collision. Only after this strong descendant certificate is available does the extractor compare coordinate siblings at the earlier folding node and obtain the relaxed incoming source. This is the one-fold step of Theorem 10.31; Theorem 10.40 iterates it backward from the terminal.

The Fiat–Shamir extractor replaces the interpolation by a test. Once the recursive extractor has authenticated the outgoing source, the exact branch relation is a decidable property of that source, and the checks between the outgoing commitment and the next fold challenge are public-coin tests of it. The extractor of Theorem 10.16 evaluates the relation on the extracted source. If the relation fails, it extracts once more below the same commitment: a different source is a matrix collision, and the same source requires a challenge in a small set fixed by that source. This uses the uniqueness condition of a strong reduction directly and needs no transcript tree for the strong checks (Proposition 10.43). In both forms, SuperNeo's framework explains the proof obligation, while Akita's interleaved order requires its own composition. Every failure of extraction uniqueness is a matrix-view kernel vector, which Lemma 10.19 converts to ordinary MSIS.

Per-level knowledge error. For a schedule sch, write $\mathsf{Lev}_{\mathsf{nt}}(\mathsf{sch})$ for its nonterminal fold levels. For level j, let $\mathcal{G}_j$ be its ordered source groups. For each $g \in \mathcal{G}_j$, let $W_{j,g}$ be its number of fold-challenge coordinates and $\mathcal{C}^{\mathsf{acc}}_{j,g}$ the exact family sampled by the verifier after every fixed, witness-independent filter. Thus $W_{j,g}$ is the number of claims in the group times its number of source blocks. Define

$$\varepsilon_{\mathsf{cw},j} := \sum_{g \in \mathcal{G}_j} \frac{W_{j,g}}{|\mathcal{C}^{\mathsf{acc}}_{j,g}|}, \qquad \varepsilon_{\mathsf{cw}}(\mathsf{sch}) := \sum_{j \in \mathsf{Lev}_{\mathsf{nt}}(\mathsf{sch})} \varepsilon_{\mathsf{cw},j} + \varepsilon_{\mathsf{term}}. \tag{219}$$

For one ordinary opening, $W_{j,g} = B_j$ and $\varepsilon_{\mathsf{cw},j} = B_j/|\mathcal{C}^{\mathsf{acc}}_j|$. Let μ'_j be the extended-witness variable count, n_j the number of rows in the extended relation matrix, and $\xi_{\mathsf{ring},j}$ the ring-reduction arity from (217) at level j. With an operator-norm filter, the denominator uses the accepted family (47). Response-search probes enter Q, without changing this fixed family. Let $\beta_j \in \{0,1\}$ indicate whether the schedule-derived binary support is nonempty at this nonterminal level. Let the range tree have stage degrees $d_{j,1}, \ldots, d_{j,S_j}$ and interstage frontier sizes $m_{j,1}, \ldots, m_{j,S_j-1}$. For every Euclidean source group g, the last range-tree stage includes its fused norm term from (122). Define

$$\nu_{j,g} := \begin{cases} 0, & \text{coefficient-}\ell_\infty\text{ route,} \\ 2, & \text{direct Euclidean route,} \\ N_{\mathsf{pair},j,g} + \delta_{f,j,g}, & \text{digit-expanded Euclidean route,} \end{cases} \qquad \nu_j := \sum_{g \in \mathcal{G}_j} \nu_{j,g}.$$

By (216), ν_j is the exact total numerator from every Stage 1 norm merge, Stage 2 shifted-power binding to the response digits, and digit-inner-product-claim batch at level j. These terms are absent at the Akita terminal, where the verifier computes the integer norm directly. The interactive knowledge error of one

scheduled nonterminal level is

$$\begin{aligned}\varepsilon_j = \underbrace{\varepsilon_{\mathsf{cw},j}}_{\text{fold CWSS}} + \frac{1}{|\mathbb{F}_{q^k}|}\Bigg(& \underbrace{\xi_{\mathsf{ring},j}}_{\text{ring reduction}} + \underbrace{\lceil \log n_j \rceil}_{\text{row batching}} + \underbrace{\mu'_j}_{\text{range point}} \\ & + \underbrace{\beta_j \mu'_j}_{\text{binary-support point}} \\ & + \underbrace{\sum_{s=1}^{S_j} \mu'_j (d_{j,s}+1)}_{\text{range-tree stages}} + \underbrace{\sum_{s=1}^{S_j-1} (m_{j,s}-1)}_{\text{range-tree frontiers}} \\ & + \underbrace{1}_{\text{relation/range batching}} + \underbrace{\beta_j}_{\text{binary batching}} + \underbrace{\nu_j}_{\text{norm-claim binding}} + \underbrace{3\mu'_j}_{\text{relation check}} \Bigg).\end{aligned} \tag{220}$$

The Akita terminal is not assigned a copy of (220). Its clear checks have the shorter layout in Proposition 10.28, and its fold-CWSS contribution is $\varepsilon_{\mathsf{term}}$ from (213).

Let $o_j \in \{0,1\}$ record whether nonterminal level j runs the Stage-3 setup product sum-check. For every offloaded edge $j \to j+1$, we have $o_j = 1$. When $o_j = 1$, let $N^R_{\mathsf{setup},j}$ be the padded setup-domain length used by Stage 3; set it to 1 otherwise. The schedule-level error is

$$\varepsilon^{\mathsf{sch}}_j := \varepsilon_j + o_j \frac{2 \log_2 N^R_{\mathsf{setup},j}}{|\mathbb{F}_{q^k}|}. \tag{221}$$

The added term accounts for the degree-2 Stage-3 sum-check. The fresh group combination at the edge's nonterminal receiver is accounted for schedule-wide below. An admissible offloaded edge cannot target the Akita terminal.

Lemma 10.39 (Setup offloading inside the scheduled reduction). *For an offloaded edge $j \to j+1$, the extra terms are the Stage 3 term in $\varepsilon^{\mathsf{sch}}_j$ and the fresh two-group combination term in $\varepsilon_{\mathsf{grp}}$ below. If $j+1 \geq 2$, its two incoming groups also contribute to the one receiver-owned tensor reduction. Offloading does not change either fold's CWSS extraction or response-norm bounds. Binding the setup opening of the registered commitment $C_{S,j}$ is covered by the per-matrix Module-SIS inventory $\mathcal{I}(\mathsf{sch})$, at the setup occurrence's comparison-class radii. The receiver $j+1$ 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 $\log_2 N^R_{\mathsf{setup},j}$ variables, giving its term in (221). At level $j+1$, the witness and setup prefix are two instances, each at its own point. At a packing receiver, their commitments, points, and claimed values are fixed before the normalized scalar in (170); two distinct scalar values extract both claims. At an evaluation-trace receiver, Lemma 10.27 first reduces them in one shared tensor reduction; it does not identify distinct incoming opening points. Only after the shared opening payload is fixed does the receiver sample the fresh evaluation-trace claim coefficients. A nonzero residual vector passes this batch with one-shot probability at most $1/|E|$, and two distinct values of the normalized scalar coefficient extract both claims. The mode-selected nonterminal multi-instance fold then handles both groups. The setup-offloading soundness lemma connects an accepted opening of the schedule-selected $C_{S,j}$ to the true setup scan. The setup opening is checked under the same scheduled matrix inventory as the other opening groups.

Let $\mathsf{EvalRecv}(\mathsf{sch})$ be the evaluation-trace receivers in the schedule, including its terminal, and retain the incoming-group notation of Definition 10.26. Define

$$\varepsilon_{\mathsf{tensor}}(\mathsf{sch}) := \sum_{h \in \mathsf{EvalRecv}(\mathsf{sch})} \varepsilon_{\mathsf{tensor},h}. \tag{222}$$

This sum includes every nonterminal evaluation-trace receiver that exists and the terminal tensor reduction, regardless of the terminal's absolute depth. Let $\mathsf{Recv}(\mathsf{sch})$ be the internal receivers whose mode-specific relation combines $N_{\mathsf{clm},h} > 1$ fixed claims, excluding application-level public batching. This includes a packing receiver with an offloaded setup claim and every evaluation-trace receiver with more than one tensor-reduction output claim. Set

$$\varepsilon_{\mathsf{grp}}(\mathsf{sch}) := \sum_{h \in \mathsf{Recv}(\mathsf{sch})} \frac{N_{\mathsf{clm},h} - 1}{|E_h|}. \tag{223}$$

At a packing receiver the normalized coefficients are sampled after its commitments, points, and claimed values are fixed. At an evaluation-trace receiver they are sampled only after the shared opening payload is fixed. The summand is its coordinate-wise extraction charge; its 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.

Let $\mathsf{Batch}(\mathsf{sch})$ contain the public application-batching steps in the schedule. For $a \in \mathsf{Batch}(\mathsf{sch})$, let Λ_a be its number of claims and E_a its challenge field, and define

$$\varepsilon_{\mathsf{batch}}(\mathsf{sch}) := \sum_{a \in \mathsf{Batch}(\mathsf{sch})} \frac{\Lambda_a - 1}{|E_a|}. \tag{224}$$

An upper bound on the interactive knowledge error of the scheduled protocol is

$$\begin{aligned} \varepsilon_{\mathsf{int}}(\mathsf{sch}) := \sum_{j \in \mathsf{Lev}_{\mathsf{nt}}(\mathsf{sch})} \varepsilon_j^{\mathsf{sch}} + \varepsilon_{\mathsf{tensor}}(\mathsf{sch}) + \varepsilon_{\mathsf{grp}}(\mathsf{sch}) \\ + \varepsilon_{\mathsf{term}} + \varepsilon_{\mathsf{batch}}(\mathsf{sch}). \end{aligned} \tag{225}$$

Each public coin is charged once in this expression. The first sum contains the nonterminal folds and their strong suffixes, including a producer's Stage 3 setup check. The next two terms contain the receiver-owned tensor reductions and their later internal group combinations. The fourth term is the terminal fold, and the last term is application batching before the root fold.

For a transcript-bound profile $(\lambda_{\mathsf{FS}}, Q_{\max})$, define the schedule's Fiat–Shamir error budget by

$$\varepsilon_{\mathsf{FS}}(\mathsf{sch}; Q_{\max}) := (Q_{\max} + 1)\varepsilon_{\mathsf{int}}(\mathsf{sch}) \le 2^{-\lambda_{\mathsf{FS}}}. \tag{226}$$

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

Security model. The Fiat–Shamir reduction below uses the classical random-oracle model, including the separately indexed programmable coordinate streams of Lemma 10.2. Coordinate-stream queries count toward the same public query budget; their lazy bit expansion is the ideal-XOF abstraction specified in Section 9.3. The reduction uses the classical Q-query checkpoint extractor of Theorem 10.16. The Module-SIS inventory is sized against a quantum lattice-attack cost model, but that parameter choice does not upgrade the random-oracle reduction to the QROM. A post-quantum knowledge-soundness claim for the noninteractive protocol requires a separate QROM Fiat–Shamir theorem for this multi-round, coordinate-wise extraction tree. That theorem is explicitly outside the scope of this paper and remains future work. 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 $\mathcal{H}_{\mathsf{FS}}$. The count includes every prover-controlled nonce probe used for response grinding. The theorem still states an explicit upper bound for the adversary's actual query count $Q \le Q_{\max}$. 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.

Extractor running time. Let $\mathsf{Rounds}(\mathsf{sch})$ list the public-coin challenge stages of the actual protocol, including every shared tensor-reduction prefix, public claim combination, fold, range stage, setup-product sum-check, and terminal. A vector sampled together remains one stage. Each stage r records its coordinate spaces $S_{r,u}$, interpolation arities $k_{r,u}$, and whether it uses a flat coordinate-wise family or a nested interpolation tree. Define

$$\mathsf{Tree}(\mathsf{sch}) := \prod_{r\in\mathsf{Rounds}(\mathsf{sch})} b_r, \qquad b_r := \begin{cases} 1+\sum_{u\in U_r}(k_{r,u}-1), & \text{flat stage,}\\ \prod_{u\in U_r} k_{r,u}, & \text{nested stage.}\end{cases} \tag{227}$$

A fold remains flat, with factor $1+\sum_g W_{j,g}$. A normalized affine combination of Λ claims has factor Λ; this applies to both tensor and evaluation-trace claim combinations. In contrast, tensor row batching has factor 2^{κ}, the range point has factor $2^{\mu'}$, and relation-row batching has factor $2^{\lceil \log_2 n_{\mathsf{rel}}\rceil}$. These are nested stages. A scalar degree-D sum-check challenge contributes $D+1$; its successive protocol rounds already supply a nested tree. In particular, the binary support check reuses the final range-stage tree rather than introducing another nested tree. Setup boundaries retain their separate internal claim-combination stages. The companion error of Lemma 10.14 is the sum of the coordinate charges already accumulated in (225). This complete tree is the input of Theorem 10.40. The Fiat–Shamir extractor branches only at the fold challenges and at the checkpoints of Proposition 10.43: the ring-reduction challenge of every nonterminal level and, when the application root combines more than one claim, its scalar-batching challenge. Let $\chi_{\mathsf{root}} \in \{0,1\}$ indicate the latter case. Define

$$\mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch}) := 2^{|\mathsf{Lev}_{\mathsf{nt}}(\mathsf{sch})|+\chi_{\mathsf{root}}}\,(1+B_{\mathsf{term}}) \prod_{j\in\mathsf{Lev}_{\mathsf{nt}}(\mathsf{sch})} \Big(1+\sum_{g\in\mathcal{G}_j} W_{j,g}\Big). \tag{228}$$

The power of two counts one additional extraction per checkpoint, $1+B_{\mathsf{term}}$ is the terminal fold factor, and the product ranges over the nonterminal fold factors. Then $\mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch}) \le \mathsf{Tree}(\mathsf{sch})$: both contain the fold factors, and the complete tree has in addition a range-point factor $2^{\mu'} \ge 2$ at every nonterminal level and a factor $\Lambda \ge 2$ at a batching root. Admissibility requires $\mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch})$ to satisfy (175). Without that condition, the extraction theorem may give a valid error expression but not an expected polynomial-time knowledge extractor. The polynomial is measured in $(\lambda, N_{\mathsf{exp,max}})$, where $N_{\mathsf{exp,max}}$ is the total explicit instance-and-witness length, not in its logarithmic variable count $\mu_{\max}$.

For a classical Fiat–Shamir adversary $\mathcal{A}$ making at most Q queries to $\mathcal{H}_{\mathsf{FS}}$, let $T_{\mathsf{FS}}(\mathcal{A}, Q, \mathsf{sch})$ denote the expected running time of the extractor of Theorem 10.16 for the checkpoint structure of Proposition 10.43. 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 fixed-filter coordinate of a fold, the extractor programs a conditional raw tape using Lemma 10.2. Define

$$T_{\mathsf{filter}}(\mathsf{sch}) := \max_{r\in\mathsf{Rounds}(\mathsf{sch})} \sum_{u\in U_r} \frac{L_{r,u}}{\underline{a}_{r,u}}, \tag{229}$$

where an unfiltered challenge has $L_{r,u} = \underline{a}_{r,u} = 1$. Each run is accompanied by polynomial-time work: the transcript checks, the decoders and relation tests at the checkpoints, and the deterministic algebraic extractors at the folds. Consequently,

$$\begin{aligned} T_{\mathsf{FS}}(\mathcal{A}, Q, \mathsf{sch}) \le{}& (Q+2)\cdot \mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch}) \\ &\cdot \mathsf{poly}\big(\lambda, |\mathsf{sch}|, N_{\mathsf{exp,max}}, T_{\mathcal{A}}, T_{\mathsf{ver}}(\mathsf{sch}), T_{\mathsf{filter}}(\mathsf{sch})\big), \end{aligned} \tag{230}$$

where the polynomial bounds the work per run and depends on the chosen classical rewinding implementation. This is an expected-time bound. Together with (175), it is polynomial in the admitted security and explicit-instance parameters.

For each matrix instance I, the constructed MSIS adversary first embeds its ordinary MSIS challenge into the I-th setup view, then runs this same FS extractor, and finally keeps the output only if the collision names matrix I. Thus

$$T_{\mathcal{B}_I} \le T_{\mathsf{FS}}(\mathcal{A}, Q, \mathsf{sch}) + T_{\mathsf{embed}}(I) + T_{\mathsf{coll}}(I), \tag{231}$$

where $T_{\mathsf{embed}}(I)$ is linear in the setup length and $T_{\mathsf{coll}}(I)$ 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 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 (204); 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 (225), with an extractor whose expected running time is polynomial in* $\mathsf{Tree}(\mathsf{sch})$*.*

This statement makes no assumption about the distribution of honest responses. A model-derived response bound enters only as a public verifier-enforced bound in an accepting transcript.

The tree of this theorem is the complete one. Proposition 10.43 gives the same error with the smaller tree of (228).

Proof. **Terminal base case.** Apply Proposition 10.28 to the terminal subtree. Its clear $\mathbf{A}$ and evaluation-trace fold-evaluation equations extract the terminal source against the predecessor-bound inner state, while its fixed trace equation binds the extracted partials to the tensor-reduction output. Applying Lemma 10.27 to the terminal's incoming recursive-witness group returns the ordinary base-field opening entering the terminal, or a terminal $\mathbf{A}$-matrix collision. Thus 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. That theorem 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 10.27 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 and certified norms. At the root, each extracted weak opening supplies, through its source projection, the algebraic decommitment of Definition 10.21. Applying $\mathsf{Dec}_{\mathsf{shape}_{\mathsf{com},j}}$ to $(\mathbf{s}_{j,c,i})_{i\in[B_j]}$ defines the corresponding decoded full commitment-grid vector. Lemma 10.33 separates the public batch and forces the multilinear extension of each decoded vector at $\mathbf{r}_j$ to equal $\phi_{j,c}v_{j,c}$. These are exactly the two conjuncts of (204); no padding-image claim is used. Every accepted coefficient route supplies the certified response envelope used in its collision inventory. On a Euclidean route, Lemma 10.34 additionally forces the public squared norm to equal the sum of squares of the reconstructed integer response; together with the certified challenge-operator bound, Corollary 10.35 gives the scheduled Euclidean collision radius. These arguments use the predicates proved in the accepting tree, not the distribution from which an honest response was modeled.

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 (225). This proves the stated interactive 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 instead returns a valid algebraic decommitment, and Corollary 10.25 makes its source projection computationally unique among successful extractions. This is exactly why Definition 3.10 states knowledge soundness using a decommitment relation rather than by re-running Commit and comparing its output state.

A stronger claim that the extractor recovers the canonical commitment state would require an additional canonicality or source-shortness argument. It does not follow from the bounded folded responses, because division by a unit need not preserve norm, and it is not supplied merely by invoking the strong–weak composition theorem. Akita does not use that stronger claim.

Checkpoint structure of a schedule. The proof of the Fiat–Shamir theorem does not build the complete tree of (227). It applies Theorem 10.16, which needs sibling transcripts only at the stages where the extractor subtracts them. We assign the stages of $\mathsf{Rounds}(\mathsf{sch})$ to the three stage types of Definition 10.15 as follows.

– *Extraction stages:* the fold challenge of every nonterminal level and of the terminal, with the flat coordinate-wise structure of Definition 10.30. Its arity is $1 + \sum_g W_{j,g}$ at level j and $1 + B_{\mathsf{term}}$ 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 5. If the application root combines $\Lambda > 1$ claims, its scalar-batching challenge ζ_{batch} is also a checkpoint. It is the first challenge of the protocol and follows the root's opening payload.
– *Plain stages:* every other stage. These are the range, norm, row-batching, and sum-check challenges of steps 5 and 6, 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. The segment of a root checkpoint is ζ_{batch} alone, because the next challenge is the root's fold challenge. Every plain stage 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; it is the outgoing source of the discussion above. 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 (106), so the exact branch relation can be evaluated on it. 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, let g_t be its work-predicate target and E its challenge field, and put $\varepsilon_{\mathsf{st},t} := 2^{-g_t}\nu_t/|E|$, where the numerator ν_t is:

– $\xi_{\mathsf{ring}} - 1$ 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.

Lemma 10.42 (Checks of a fixed successor source). *Fix an admissible schedule and a nonterminal level h. For every successor source σ and every stage input u from the input of the level's α through the input of the next fold challenge, there is a bit $D(u;\sigma)$, and for every stage t of the segment a set $\mathsf{Bad}_t(u;\sigma)$ of answers, such that:*

(i) *at the input of α, $D(u;\sigma) = 0$ exactly when no verifier check determined by u fails and $\mathbf{w}_\sigma$ satisfies the exact branch relation of u;*
(ii) *at the input of the next fold challenge, $D(u;\sigma) = 0$ exactly when no verifier check determined by u 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 u equals $\mathbf{a}_{\mathsf{term}}^\top \mathbf{G}_{b_{\mathsf{term}},M_{\mathsf{term}}}\mathbf{s}_i$;*
(iii) *if $D(u;\sigma) = 1$ at a stage-t input and the answer ρ is outside $\mathsf{Bad}_t(u;\sigma)$, then $D(u';\sigma) = 1$ at every next input u' that extends u by ρ; and*
(iv) *$\mathsf{Bad}_t(u;\sigma)$ has probability at most $\varepsilon_{\mathsf{st},t}$.*

The numbers ν_t of the segment are termwise at most the numerators that (220), (221), $\varepsilon_{\mathsf{tensor},h+1}$, and (223) charge to the same challenges. The binary-support point term $\beta_j\mu'_j$ of (220) has no counterpart.

Proof. Associate with every input u its *pending claims*: the claims that the protocol has stated or derived by u and has not yet checked. Every table in them is computed from σ, $\mathbf{S}^\flat$, and the public data in u. A batching

coefficient that has not yet been 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)–(iii) of the exact branch relation. 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. Put $D(u;\sigma) = 1$ if a verifier check determined by u fails or a pending claim is false. A failed check stays failed on every extension, and then $\mathsf{Bad}_t(u;\sigma)$ is empty. Otherwise fix an order on the pending claims and let $\mathsf{Bad}_t(u;\sigma)$ be the set described below for the first false one. Keeping $D = 1$ requires only that some pending claim remain false, so the bad set of a stage protects one false claim and not every false claim. This gives (i) and (ii). 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 $\xi_{\mathsf{ring}} - 1$ values of α are exceptional, by (217), 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 (112), 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 ν_t: one less than the number of claims in the frontier and digit-pair batches, the positive degree in η_{norm} noted after (125), 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 $\sum_{x\in\mathcal{I}_{\mathsf{bin}}} \widetilde{\mathsf{eq}}(\mathbf{r}, x)\, \mathbf{w}_\sigma(x)(\mathbf{w}_\sigma(x)+1)$, a multilinear polynomial in the coordinates of $\mathbf{r}$ 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. Consider a round of that sum-check. If its sum-check claim is false, the bad set is the one given above for sum-check rounds, and the false claim of the next round keeps $D = 1$ whatever happens to the residual. If its sum-check claim is true and the residual is nonzero, the bad set is this one value. In both cases the bad set has at most ν_t elements, so the round is charged once. For this reason the lemma has no term for the binary-support point.

Integer norms. Suppose that no verifier check on u fails, that condition (i) of the exact branch relation holds, and that condition (iii) fails. On the direct route, both sides of (117) are integers in $[0, q)$, as in the proof of Lemma 6.3, so the field identity tested by the norm term is false. On the digit-expanded route, some claim $p_{t,h,k}$ 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 $\mathcal{E}_{\mathsf{int}}(\hat{\mathbf{z}})$, and the passing comparison with the decoded $\mathcal{E}_{\mathsf{resp}}$ would give condition (iii) of the exact branch relation.

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 (169); 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. 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.

The comparison with these error terms is termwise: $\xi_{\mathsf{ring}} - 1$ against ξ_{ring}, one-shot batching bounds against coordinate-wise charges, and equal numbers elsewhere.

Proposition 10.43 (Checkpoint structure of an admissible schedule). *Fix an admissible schedule under the idealized setup. With the stage types above, its protocol has a checkpoint structure in the sense of*

Definition 10.15 for $\mathsf{R} = \mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{batch}}$ *and for the binding relation whose witnesses are the nonzero kernel vectors of a matrix view* $I \in \mathcal{I}(\mathsf{sch})$ *of norm at most* η_I*. Its extraction stages have the errors* $\varepsilon_{\mathsf{cw},j}$ *and* $\varepsilon_{\mathsf{term}}$*. Its other stages have the errors* $\varepsilon_{\mathsf{st},t}$ *of Lemma 10.42, and* $(\Lambda-1)/|E|$ *at a root checkpoint. Its tree size is* $\mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch})$ *from* (228)*, and*

$$\sum_{r\in X} e_r + \sum_{r\in H\cup N} \varepsilon_r \le \varepsilon_{\mathsf{int}}(\mathsf{sch}). \tag{232}$$

Proof. We specify the data of Definition 10.15.

Certificates. Consider a stage after the checkpoint of level $h-1$ (or after the root checkpoint) 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 (Definition 10.22) of every incoming group of level h for this transcript; in it, 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. 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 10.15. The maps below preserve the condition on verifier checks, because the transcript of their output is the common prefix of the transcripts of their inputs.

Extraction stages. At the terminal fold, apply the subtraction step in the first paragraph of the proof of Proposition 10.28 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. At a nonterminal fold, the children are certificates of the checkpoint that follows it, 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.

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. 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 10.28 gives one. Lemma 10.42 supplies D and Bad. Condition (a) is its item (i), 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).

At a root checkpoint, the candidate is the source projection of all root groups, and the test is that every decoded vector has its padded claimed value. Let $D = 1$ before ζ_{batch} if a padded claim is false for the candidate, and after it if the scalar row is false. Lemma 10.33 bounds the probability of the bad set by $(\Lambda-1)/|E|$, a weak opening satisfies the scalar row, and binding is again Corollary 10.25.

Witness. A certificate at the first stage is a weak opening of the root groups whose decoded vectors have their padded claimed values, by the root test or, for one claim, by the scalar row itself. Its source projection and decoded vectors satisfy $\mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{batch}}$, as in the last step of the proof of Theorem 10.40.

Size and errors. There are $|\mathsf{Lev}_{\mathsf{nt}}| + \chi_{\mathsf{root}}$ checkpoints, and the arities of the extraction stages are the fold factors, which gives (228). The errors of the extraction stages sum to $\varepsilon_{\mathsf{cw}}(\mathsf{sch})$. By the last part of Lemma 10.42, the remaining errors are at most the remaining terms of (225).

Theorem 10.44 (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. Let the statistical distance of the seed expansion from a uniform flat setup vector be*

$$\Delta_{\mathsf{setup}} \le N_{\mathsf{setup}}\, q\, 2^{-w}$$

as bounded in (60)*. For every classical random-oracle prover making* $Q \le Q_{\max}$ *queries to* $\mathcal{H}_{\mathsf{FS}}$*, there is an expected polynomial-time extractor and, for each* $I \in \mathcal{I}(\mathsf{sch})$*, an ordinary Module-SIS adversary* $\mathcal{B}_I$ *with running time bounded by* (231)*. By admissibility, the recorded error upper bound satisfies* (226)*.*

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 satisfies adaptive batch knowledge soundness in the classical random-oracle model for the relation* $\mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{batch}}$ *of* (204)*. More precisely, in the batch analogue of the same-execution knowledge experiment of Definition 3.10, with the declared statement* $\mathsf{x}_{\mathsf{batch}}$ *and Akita proof in place of one evaluation claim, the probability that the verifier accepts but the extractor returns* $\perp$ *or a witness outside this relation is at most*

$$(Q+1)\,\varepsilon_{\mathsf{int}}(\mathsf{sch}) + \Delta_{\mathsf{setup}} + \sum_{I\in\mathcal{I}(\mathsf{sch})} \mathsf{Adv}_{\mathsf{MSIS}^{(p_I)}_{q,d_I}(n_I,m_I,\eta_I)}(\mathcal{B}_I). \tag{233}$$

This is an explicit reduction bound for the adversary's query count Q*. By* (226)*, the right-hand side of* (233) *is at most*

$$2^{-\lambda_{\mathsf{FS}}} + \Delta_{\mathsf{setup}} + \sum_{I\in\mathcal{I}(\mathsf{sch})} \mathsf{Adv}_{\mathsf{MSIS}^{(p_I)}_{q,d_I}(n_I,m_I,\eta_I)}(\mathcal{B}_I).$$

The first term of (233) *can be replaced by the smaller quantity*

$$(Q+1)\,\varepsilon_{\mathsf{cw}}(\mathsf{sch}) + \sum_t \big(\mathbb{E}[Q_t]+1\big)\,\varepsilon_{\mathsf{st},t}, \tag{234}$$

where t *ranges over the checkpoint and plain stages of Proposition 10.43,* $\varepsilon_{\mathsf{st},t}$ *is the error of stage* t *stated there, and* Q_t *is the number of stage-*t *inputs among the prover's queries, so that* $\sum_t Q_t \le Q$*. None of these bounds assumes that honest responses follow the response model. The model may select a public bound, but knowledge soundness uses only that bound and the digit range enforced by the verifier in an accepting transcript. Queries used to search for such a transcript are included in* Q*.*

Proof. We prove the statement by a sequence of reductions.

Step 1: idealize the setup. Replace the concrete XOF expansion of Equation (60) by a uniform flat vector $\mathbf{S} \leftarrow \mathbb{Z}_q^{N_{\mathsf{setup}}}$. By the statistical-distance bound in Section 4.3, this changes the adversary's success probability by at most Δ_{setup}. The setup-expansion and Fiat–Shamir inputs occupy disjoint domain-separation namespaces. In the ideal experiment the simulator lazily answers every setup-domain query from this sampled vector, including page queries, while leaving the Fiat–Shamir namespace as an independent random oracle. Thus an adversary that recomputes the public setup sees a consistent oracle; the coefficient-sampling bias is the only distributional change. All matrices and setup data used below are deterministic functions of this single uniform vector. Applying this deterministic map cannot increase the statistical distance.

Step 2: a checkpoint structure. Consider the public-coin interactive protocol specified by the schedule, with challenge rounds listed in $\mathsf{Rounds}(\mathsf{sch})$. By Proposition 10.43, it has a checkpoint structure whose extraction stages are the fold challenges, whose tree size is $\mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch})$, and whose stage errors sum to at most $\varepsilon_{\mathsf{int}}(\mathsf{sch})$ by (232). For every fixed-filtered fold coordinate, Lemma 10.2 supplies an exact conditional random-oracle tape with overhead bounded by (229); response-dependent nonce probes remain part of Q. This is the sampling hypothesis of Theorem 10.16. No other stage is programmed on a prefix of its answer.

Step 3: extract. Apply Theorem 10.16. By (191), the probability that the original proof accepts but the extractor returns neither a witness for $\mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{batch}}$ nor a short kernel vector for a matrix view in $\mathcal{I}(\mathsf{sch})$ is at most (234), which is at most $(Q+1)\varepsilon_{\mathsf{int}}(\mathsf{sch})$ by (192) and (232). The extractor runs the prover at most $(Q+2)\,\mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch})$ times in expectation, which gives (230). The response-model hypothesis is absent from this step.

Step 4: reduce shared-view collisions to ordinary MSIS. If the extractor outputs a short kernel vector for a matrix view of the uniform setup vector, apply Lemma 10.19. This constructs ordinary MSIS adversaries $(\mathcal{B}_I)_{I\in\mathcal{I}(\mathsf{sch})}$ whose total success probability is bounded by the final sum in (233). For each fixed matrix I, the adversary $\mathcal{B}_I$ embeds its ordinary MSIS challenge into the I-view, samples the rest of the shared setup uniformly, runs the same extractor, and returns the collision only when the collision matrix is I. This construction gives the running-time bound (231). Adding the losses of Steps 1, 3, and 4 gives (233).

Recomputing the last terminal block. The theorem above treats the terminal that sends every block $\mathbf{t}_i$ and every partial e_i. The protocol of Section 8.2 replaces the last of each by a digest and lets the verifier recompute it from (164). We write Π for the Fiat–Shamir protocol of Theorem 10.44 and Π_{dig} for this variant. The two differ only in these two messages and in the verifier's recomputation. We model $\mathcal{H}_{\mathsf{dig}}$ as a random oracle with

256-bit output, independent of $\mathcal{H}_{\mathsf{FS}}$ and $\mathcal{H}_{\mathsf{setup}}$. Its input encoding is injective: field elements are encoded canonically and the schedule fixes the block length. The Fiat–Shamir transcript encoding is also fixed by the schedule, so a string determines whether it is a transcript prefix of Π_{dig} and, if so, where its digest positions lie.

Lemma 10.45 (Recomputing the last terminal block). *Let P_{dig} be a classical prover for Π_{dig} that makes at most Q queries to $\mathcal{H}_{\mathsf{FS}}$ and at most Q_{dig} queries to $\mathcal{H}_{\mathsf{dig}}$, where both counts include the queries made by the verifier on the prover's output. There is a prover P for Π that makes at most Q queries to $\mathcal{H}_{\mathsf{FS}}$, runs P_{dig} once with $O(Q + Q_{\mathsf{dig}})$ additional table operations, and outputs the same statement, such that*

$$\Pr[\Pi_{\mathsf{dig}} \text{ accepts } P_{\mathsf{dig}} \text{ and } \Pi \text{ rejects } P] \le \varepsilon_{\mathsf{dig}}(Q, Q_{\mathsf{dig}}) := \frac{\binom{Q_{\mathsf{dig}}}{2} + 2Q\,Q_{\mathsf{dig}}}{2^{256}}. \tag{235}$$

Consequently Theorem 10.44 holds for Π_{dig} with $\varepsilon_{\mathsf{dig}}(Q, Q_{\mathsf{dig}})$ added to (233), where $Q \le Q_{\max}$ and Q_{dig} are counted as in this lemma. This term is not multiplied by $Q + 1$.

Proof. The prover P runs P_{dig}, answers its $\mathcal{H}_{\mathsf{dig}}$ queries by lazy sampling, and records them in a table. A transcript prefix of Π_{dig} contains at most two digest positions. When P_{dig} queries $\mathcal{H}_{\mathsf{FS}}$ on such a prefix, P looks up each digest in the table. If every digest has a recorded preimage, P replaces each digest by its preimage, which yields a transcript prefix of Π, forwards that prefix to its own oracle, and returns the answer. On every other query, namely a prefix with an unopened digest or a string that is not a transcript prefix of Π_{dig}, P returns a fresh uniform answer and repeats it on a repeated query. When P_{dig} halts, P runs the Π_{dig} verifier on its output under the same rules, and outputs the proof with the two recomputed blocks in place of the digests.

Consider two events. In the first, two distinct $\mathcal{H}_{\mathsf{dig}}$ queries return the same value; this has probability at most $\binom{Q_{\mathsf{dig}}}{2}2^{-256}$. In the second, a fresh $\mathcal{H}_{\mathsf{dig}}$ query returns a string that already occupies a digest position of an earlier $\mathcal{H}_{\mathsf{FS}}$ query. At most $2Q$ such strings exist, so this has probability at most $2Q\,Q_{\mathsf{dig}}2^{-256}$. Both events range over the verifier's queries as well, which is why the lemma counts them.

Suppose neither event occurs. Then a digest has at most one recorded preimage, and a digest that was unopened when it appeared in an $\mathcal{H}_{\mathsf{FS}}$ query is never opened later. The replacement of digests by preimages is therefore a fixed injective map 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_{dig} are distributed as a random oracle. If the Π_{dig} verifier accepts, its two recomputed blocks hash to the two digests in the proof. The verifier's $\mathcal{H}_{\mathsf{dig}}$ queries on these blocks follow its $\mathcal{H}_{\mathsf{FS}}$ queries, so by the second event each digest was already opened before the first $\mathcal{H}_{\mathsf{FS}}$ query that contains it, and its unique recorded preimage is the recomputed block. Every challenge of the accepted proof thus equals the answer of $\mathcal{H}_{\mathsf{FS}}$ on the corresponding prefix of the output of P. The recomputed blocks satisfy (161) and (162) by (164), and the remaining checks of the two verifiers coincide. Hence Π accepts the output of P.

For the final claim, apply the extractor of Theorem 10.44 to P. An execution in which Π_{dig} accepts and extraction fails is either an execution in which Π accepts and extraction fails, or one counted in (235). The reduction records the oracle queries of P_{dig}, so it is a classical random-oracle argument, in line with the security model above.

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, which is why the replacement map in the proof preserves the order of messages and challenges. The transcript absorbs different bytes for a digest than for a block, so the challenges of a Π_{dig} proof differ from those of a Π proof for the same witness; the lemma relates the two provers and does not claim that proofs can be converted after the fact.

For one commitment containing one decoded full-grid vector, with one opening claim and no natural-domain padding obligation, $\mathsf{Rel}^{\mathsf{Akita}}_{\mathsf{batch}}$ specializes to $\mathsf{Rel}_{\mathsf{eval}}$ in (51). If a surrounding application separately certifies that an extracted common-grid vector is the canonical padding of a smaller evaluation vector, then (75) additionally 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 E.1; no global zero-padding predicate is then needed. Completeness and binding are separate properties of the PCS definition and are not asserted by Theorem 10.44.

Corollary 10.46 (Schedule-specific Fiat–Shamir ceiling). *For a fixed schedule and query budget, define its effective Fiat–Shamir strength by*

$$\lambda_{\mathsf{FS}}^{\mathsf{eff}}(\mathsf{sch}; Q_{\max}) := -\log_2((Q_{\max}+1)\varepsilon_{\mathsf{int}}(\mathsf{sch}))\,. \tag{236}$$

An admissible schedule may claim only $\lambda_{\mathsf{FS}} \le \lambda_{\mathsf{FS}}^{\mathsf{eff}}$. If its interactive error contains a nonzero term $c/|E|$, then

$$\lambda_{\mathsf{FS}}^{\mathsf{eff}} \le \log_2 |E| - \log_2 c - \log_2(Q_{\max}+1). \tag{237}$$

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 (226) after taking negative base-two logarithms. Since $\varepsilon_{\mathsf{int}}(\mathsf{sch}) \ge c/|E|$, the same calculation gives (237).

Corollary 10.47 (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 $\mathsf{Rel}_{\mathsf{batch}}^{\mathsf{Akita}}$ nor a short nonzero kernel vector for some $I \in \mathcal{I}(\mathsf{sch})$ is at most $(Q+1)\varepsilon_{\mathsf{int}}(\mathsf{sch})$. Replacing the uniform setup by the concrete seed expansion adds at most Δ_{setup}.*

10.3 Schedules Chosen from a Public Family

The preceding theorems fix one schedule before analyzing the protocol. Some applications, however, learn the lengths of committed evaluation vectors only when a proof is produced. We therefore allow the prover to declare these lengths and then choose among a public family of schedules. The security question is not how the schedule was found. It is whether the verifier binds the choice before the first challenge and whether every schedule it may accept satisfies the same admissibility and security conditions.

Let G_0 be the number of application-supplied groups at the root, and write the declared evaluation-vector lengths as

$$\vec{N} = (N_0, \dots, N_{G_0-1}).$$

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, and define the complete accepted family by

$$\mathcal{S}_{\mathcal{P}} := \bigcup_{\vec{N}} \mathcal{S}_{\mathcal{P}}(\vec{N}),$$

where the union ranges over admitted declarations. The transcript binds $(\vec{N}, \mathsf{sch})$, together with the common $(\lambda_{\mathsf{FS}}, Q_{\max})$ profile, before the first protocol challenge.

There are two natural ways to specify $\mathcal{S}_{\mathcal{P}}(\vec{N})$. A public deterministic planner, with fixed tie breaking, defines the canonical schedule

$$\mathsf{sch} = \mathsf{Plan}_{\mathcal{P}}(\vec{N}), \tag{238}$$

in which case the accepted set is a singleton. Alternatively, the prover may transmit a schedule and the verifier may accept any schedule approved by the independent validator within the finite policy. The first approach asks the verifier to recover a prescribed output; the second asks it to check admissibility. Neither approach makes the planner's search part of proof verification.

We first lift completeness from one fixed schedule to the accepted family. The resulting statement preserves the status of the response bounds used to select each schedule.

Corollary 10.48 (Size-adaptive completeness). *Suppose that every schedule in $\mathcal{S}_{\mathcal{P}}$ satisfies the hypotheses of Theorem 10.9, and set*

$$\varepsilon_{\mathsf{comp}}^{\mathcal{P}} := \max_{\mathsf{sch} \in \mathcal{S}_{\mathcal{P}}} \varepsilon_{\mathsf{comp}}(\mathsf{sch}). \tag{239}$$

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 $1 - \varepsilon_{\mathsf{comp}}^{\mathcal{P}}$.

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 (184) 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 (239). The same conditioning shows why a model-derived bound must hold uniformly over the reachable prefixes of every schedule in the accepted family.

Knowledge soundness requires one further piece of accounting. A prover that chooses among accepted schedules may encounter different commitment matrices. We collect all of these Module-SIS instances before stating the reduction:

$$\mathbb{I}_{\mathcal{P}} := \bigcup_{\mathsf{sch}\in\mathcal{S}_{\mathcal{P}}} \mathcal{I}(\mathsf{sch}).$$

Because the public policy has finite choice sets and bounded recursion depth, it induces a finite family $\mathbb{I}_{\mathcal{P}}$ of Module-SIS instances.

We can now apply the fixed-schedule extractor without conditioning the public setup on the prover's schedule choice. This is precisely the role of the adaptive shared-prefix reduction in Corollary 10.20.

Theorem 10.49 (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 (226) and (175) for the common public profile.*

Then the Fiat–Shamir batched opening with declared sizes is knowledge sound in the sense of Theorem 10.44. 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 (204); applications that use only restrictions to the application's used slots may instead apply Lemma E.1.

For a prover making $Q \le Q_{\max}$ queries to $\mathcal{H}_{\mathsf{FS}}$, extraction fails with probability at most

$$(Q+1)\max_{\mathsf{sch}\in\mathcal{S}_{\mathcal{P}}} \varepsilon_{\mathsf{int}}(\mathsf{sch}) + \Delta_{\mathsf{setup}} + \sum_{I\in\mathbb{I}_{\mathcal{P}}} \mathsf{Adv}_{\mathsf{MSIS}^{(p_I)}_{q,d_I}(n_I,m_I,\eta_I)}(\mathcal{B}_I). \tag{240}$$

The first term is at most $2^{-\lambda_{\mathsf{FS}}}$ because every accepted schedule satisfies (226). 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. 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 Proposition 10.43 gives a checkpoint structure for each schedule in $\mathcal{S}_{\mathcal{P}}$. Apply Theorem 10.16 in the form of Remark 10.18: 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 $Q+1$ times the largest sum of stage errors, which is at most the maximum in (240) by (232). The expected number of prover runs is at most $(Q+2)\max_{\mathsf{sch}} \mathsf{Tree}_{\mathsf{ck}}(\mathsf{sch})$, and the common bound (175) makes the expected running time polynomial.

Any remaining extraction failure produces a collision for an instance in $\mathbb{I}_{\mathcal{P}}$. Apply Corollary 10.20 to the prover's adaptive schedule choice. That reduction runs the choice on an unconditionally uniform flat setup vector and never conditions the setup on the selected schedule. It gives the final sum in (240); the term Δ_{setup} is the cost of the initial setup replacement.

The security claim must account for the whole accepted family. For example, if $|\mathbb{I}_{\mathcal{P}}| \le I_{\max}$ and every instance in the union has advantage at most $2^{-\lambda_{\mathsf{inst}}}$, then the final term of (240) is at most $I_{\max}2^{-\lambda_{\mathsf{inst}}}$. A concrete claim may instead sum the advantage bounds for the distinct instances, as in Appendix C. In both cases, the accounting covers the schedules that the verifier may accept, independently of how the planner orders them.

11 Response Chunking for Distributed Proving

Akita already parallelizes the arithmetic of one fold across the cores and accelerators of a single machine. We need a different protocol when the polynomial itself no longer fits in one machine's memory. In this setting, several machines hold consecutive parts of the source witness, and we want each part, together with the large objects derived from it, to remain on the machine that stores it.

Most of the fold is compatible with this storage model. Commitment images, evaluation contributions, and ring-relation numerators decompose additively across the stored pieces. Sum-check also admits distributed round generation, with the ownership and boundary exchanges specified below. This is the same basic strategy used by 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 directly. Response chunking is the protocol change that resolves this obstruction in Akita.

We first explain why the response cannot be aggregated without substantial communication. We then define the chunked witness, derive 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 states the completeness and weak-binding bounds for the chunked relation.

11.1 The Distributed-Response Obstruction

For clarity, we begin with one polynomial and one opening claim. The multi-group relation of Section 5.6 applies the same construction to each group while preserving the group order. Let the source contain N ring elements, choose $M := 2^{r_{\mathsf{pos}}}$ positions per block, and set

$$B := \lceil N/M \rceil . \tag{241}$$

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, we decompose each block into $\mathbf{s}_i$, compute its inner image $\mathbf{t}_i = \mathbf{A}\mathbf{s}_i$, and form the 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 decompose additively across these intervals. The folded response does not become short when its source interval does. The ordinary fold forms

$$\mathbf{z} = \sum_{i\in[B]} c_i^A \mathbf{s}_i, \tag{242}$$

where $c_i^A = \iota(c_i)$ under coefficient packing and $c_i^A = c_i$ under evaluation trace. Machine P_j can compute only the corresponding partial response

$$\mathbf{z}^{(j)} := \sum_{i\in\mathcal{I}_j} c_i^A \mathbf{s}_i. \tag{243}$$

Although P_j holds only its share of the source blocks, $\mathbf{z}^{(j)}$ occupies the complete ambient response space. At a balanced first fold this space is itself approximately square-root sized. Recovering (242) would therefore require an all-reduce of several full-width responses.

We cannot avoid this communication by decomposing first and adding later. Each machine can compute a balanced digit decomposition $\hat{\mathbf{z}}^{(j)}$ of $\mathbf{z}^{(j)}$, but carries prevent $\sum_j \hat{\mathbf{z}}^{(j)}$ from being a digit decomposition of $\sum_j \mathbf{z}^{(j)}$. Thus the machines cannot independently construct the ordinary recursive response segment. We instead retain their 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*. We first describe the case where source ownership follows whole blocks; below we extend it to the contiguous intervals produced by recursion. For $j \in [N_{\mathsf{chunk}}]$, chunk j corresponds to the canonical block interval

$$\mathcal{I}_j := \left[\left\lfloor \frac{jB}{N_{\mathsf{chunk}}} \right\rfloor , \left\lfloor \frac{(j+1)B}{N_{\mathsf{chunk}}} \right\rfloor \right). \tag{244}$$

The admitted values are $N_{\mathsf{chunk}} = 1$ and powers of two at most 64. The intervals are ordered, contiguous, and differ in length by at most one. When $N_{\mathsf{chunk}} > B$, some intervals are empty; the corresponding chunks remain in the authenticated layout and carry a zero honest response. The proportional boundaries also make every finer admitted partition refine every coarser one.

The schedule fixes N_{chunk} before the fold challenges, and the transcript binds the resulting witness layout. This parameter records only the number of ordered response segments, not the assignment of machines: one machine may process several chunks, or several devices may cooperate on one chunk, without changing the proof.

For each interval, the prover computes and decomposes the local response of (243). It also retains the opening and inner image digits belonging to the blocks in that interval. Chunk j therefore contributes the contiguous witness unit

$$\mathbf{w}_j := [\hat{\mathbf{z}}^{(j)} \mid \hat{\mathbf{e}}^{(j)} \mid \hat{\mathbf{t}}^{(j)}]. \tag{245}$$

The three segments behave differently. The vectors $\hat{\mathbf{e}}^{(j)}$ and $\hat{\mathbf{t}}^{(j)}$ contain only the blocks in $\mathcal{I}_j$, so their concatenations recover the ordinary complete $\hat{\mathbf{e}}$ and $\hat{\mathbf{t}}$ segments. By contrast, every $\hat{\mathbf{z}}^{(j)}$ is response-sized. Ignore the shared compression and ring-check witnesses, and suppose every chunk uses the same scheduled response-digit depth as the unchunked response $\hat{\mathbf{z}}$. The complete body then has length

$$|\mathbf{w}_{\mathsf{chunk}}| = N_{\mathsf{chunk}}\,|\hat{\mathbf{z}}| + |\hat{\mathbf{e}}| + \left|\hat{\mathbf{t}}\right|. \tag{246}$$

Response chunking therefore adds exactly $(N_{\mathsf{chunk}} - 1)|\hat{\mathbf{z}}|$ coordinates before the shared tail.

All chunks use one fold-challenge vector and one grinding nonce. Each chunk accepts the nonce only if its response satisfies the scheduled bound. The machines may test a nonce in parallel, but they cannot select different nonces for different chunks. This keeps every retained response tied to one transcript challenge.

The algebra below is independent of the opening method. Our admitted schedules use $N_{\mathsf{chunk}} > 1$ only in the leading coefficient-packing folds, where the source witness is largest, and use the coefficient-ℓ_∞ response route there. Once the recursive witness is small enough to consolidate, the schedule sets $N_{\mathsf{chunk}} = 1$; later evaluation-trace folds may then use the selective Euclidean route of Section 6.2.

11.3 One Relation over Chunked Witness Columns

We now need to check an Akita fold using the separately stored responses $\mathbf{z}^{(j)}$. Both ordinary response equations are linear in $\mathbf{z}$, so we substitute $\sum_j \mathbf{z}^{(j)}$ and distribute the equations across chunks. Adding their short images recovers the ordinary fold identities without collecting the full response on one machine.

We begin with the fold-evaluation equation. For either opening method, let $\mathcal{P}_{\mathsf{fold}}$ be the linear map that evaluates one recomposed response:

$$\mathcal{P}_{\mathsf{fold}}(\mathbf{z}) := \begin{cases} \mathcal{L}(\mathbf{G}_{b,M}\mathbf{z}), & \mathsf{OpeningMethod} = \mathsf{SubringCoefficientPacking}, \\ \mathbf{a}^\top \mathbf{G}_{b,M}\mathbf{z}, & \mathsf{OpeningMethod} = \mathsf{EvaluationTrace}. \end{cases} \tag{247}$$

The first line is valued in the coefficient-packing carrier ring, while the second is valued in the ambient ring. Since this map is linear, the chunked fold-evaluation row is the single aggregate identity

$$\sum_{j \in [N_{\mathsf{chunk}}]} \left(\sum_{i \in \mathcal{I}_j} c_i e_i - \mathcal{P}_{\mathsf{fold}}(\mathbf{z}^{(j)}) \right) = 0. \tag{248}$$

Here c_i is interpreted in the challenge subring under coefficient packing and in the ambient ring under evaluation trace, exactly as in Figure 6.

The ambient inner-consistency rows aggregate in the same way. For every $a \in [n_A]$, they assert

$$\sum_{j \in [N_{\mathsf{chunk}}]} \left([\mathbf{A}\mathbf{z}^{(j)}]_a - \sum_{i \in \mathcal{I}_j} c_i^A [\mathbf{t}_i]_a \right) = 0. \tag{249}$$

These rows constrain the sum of the retained responses. They do not separately establish the local fold identity of each chunk. The aggregate constraint is what the global opening claim needs; in the extraction argument, we accordingly bound differences between aggregate responses (Section 11.6).

The commitment equations use the opening and inner-image digits already assigned to each chunk. Only the outer $\mathbf{B}$ commitment is sliced; the opening commitment uses one unsliced $\mathbf{D}$ map. Let $\mathbf{x}_{B,s}^{(j)}$ be the contribution of chunk j to outer slice s, and let $\mathbf{D}^{(j)}$ denote the columns of $\mathbf{D}$ acting on $\hat{\mathbf{e}}^{(j)}$. The two commitment relations are

$$\mathbf{u}_s = \sum_{j \in [N_{\mathsf{chunk}}]} \mathbf{B}\mathbf{x}_{B,s}^{(j)} \qquad (s \in [S_B]), \tag{250}$$

$$\mathbf{v} = \sum_{j \in [N_{\mathsf{chunk}}]} \mathbf{D}^{(j)} \hat{\mathbf{e}}^{(j)}. \tag{251}$$

A single $\mathbf{F}$ chain binds the complete stack $\mathbf{u} = (\mathbf{u}_s)_{s \in [S_B]}$, and a single $\mathbf{H}$ chain binds $\mathbf{v}$. Their intermediate digit vectors and public payloads are shared across all chunks.

Finally, the scalar opening row 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

$$\sum_{j \in [N_{\mathsf{chunk}}]} \left(\boldsymbol{\omega}_{\mathsf{open}}^{(j)}\right)^{\top} \hat{\mathbf{e}}^{(j)} = v_{\mathsf{open}}, \tag{252}$$

where $v_{\mathsf{open}} = v$ under coefficient packing and $v_{\mathsf{open}} = \bar{v}$ after the tensor reduction under evaluation trace.

These equations can be read as a horizontal concatenation of relation matrices. If $\mathbf{M}_j$ contains the columns acting on $\mathbf{w}_j$, then the complete relation has the form

$$\mathbf{M} = [\mathbf{M}_0 \mid \mathbf{M}_1 \mid \cdots \mid \mathbf{M}_{N_{\mathsf{chunk}}-1} \mid \mathbf{M}_{\mathsf{shared}}]. \tag{253}$$

The shared columns hold the compression and ring-check tails. Every chunk contributes to the original row families and their original public targets. Consequently, quotient lifting introduces one quotient for each ordinary ring row, not one quotient per chunk; quotient-free checking introduces no quotient for any of them. The $\mathbf{F}/\mathbf{H}$ compression rows likewise occur once.

11.4 Local Images and the Recursive Commitment

The aggregate equations tell us which contributions must add up. We next explain how each machine computes its contribution using the data it holds.

There are two partitions to keep track of on the $\mathbf{B}$ side. The outer commitment was formed using S_B slices, whose boundaries are already fixed. The distributed response computation uses N_{chunk} chunks, with chunk j holding the blocks in $\mathcal{I}_j$. These partitions serve different purposes, so their boundaries need not coincide. For $s \in [S_B]$, the committed outer slice is

$$I_{B,s} = \left[\left\lfloor \frac{sB}{S_B} \right\rfloor, \left\lfloor \frac{(s+1)B}{S_B} \right\rfloor \right). \tag{254}$$

The blocks of this slice may belong to several chunks. Chunk j contributes exactly those blocks that also lie in its own interval, namely

$$\mathcal{J}_{j,s} := \mathcal{I}_j \cap I_{B,s}. \tag{255}$$

It places the corresponding $\hat{\mathbf{t}}$ 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 adding these vectors recovers the fixed slice input:

$$\mathbf{x}_{B,s} = \sum_{j \in [N_{\mathsf{chunk}}]} \mathbf{x}_{B,s}^{(j)}. \tag{256}$$

Applying $\mathbf{B}$ recovers (250) using the commitment's original columns and slice partition.

For the unsliced $\mathbf{D}$ map, let $\mathrm{inc}_{D,j}$ embed chunk j's opening digits into their canonical positions in $\hat{\mathbf{e}}$, so that $\mathbf{D}^{(j)} = \mathbf{D}\,\mathrm{inc}_{D,j}$. Each machine computes the short image $\mathbf{v}^{(j)} := \mathbf{D}^{(j)}\hat{\mathbf{e}}^{(j)}$; their sum gives (251). Each machine also computes its contribution to the scalar opening row.

Ring reduction preserves this aggregation. On the quotient-lift route, the machines add their contributions to each ordinary row numerator; the aggregator subtracts the one shared target and forms the one quotient

for that row. Equivalently, the prover may distribute an agreed additive share of each target before forming quotient contributions by ordinary polynomial division. Individual remainders need not vanish; for an honest row their sum vanishes, so the local quotients sum to the global quotient. On the quotient-free route, the same local products are added directly under the transpose-convolution weights.

At this point, the machines hold their local witness units, and the compression and ring checks supply a shared tail. To continue the recursion, we must commit to all these pieces in one fixed order. Using the quotient and compression segments of Section 5.5, the complete next witness is

$$\mathbf{w}' = \left(\Big\| _{j\in[N_{\mathsf{chunk}}]} [\hat{\mathbf{z}}^{(j)} \mid \hat{\mathbf{e}}^{(j)} \mid \hat{\mathbf{t}}^{(j)}] \right) \| \hat{\mathbf{Q}}_{\mathsf{main}} \Big\| \Big\|_{a=1}^{L_{\mathsf{comp}}} [\hat{\mathbf{c}}_a \mid \hat{\mathbf{Q}}_a], \tag{257}$$

with all quotient segments absent on the quotient-free route and the scheduled zero alignment of Section 5.5 understood. For several groups, each chunk contains their units in the fixed group order. This ordering is committed by the next level.

No machine needs to materialize all of $\mathbf{w}'$. Each machine supplies the contribution of its contiguous chunk unit to the target commitment, while an aggregation machine supplies the shared tail. The target commitment's own frozen outer slices determine where these contributions are added. The verifier still receives one commitment payload and one opening claim, exactly as in the single-chunk recursive interface.

To carry this ownership into the next fold, we keep each chunk unit on its machine and assign the shared tail to the last machine. We fix ownership before sampling the next fold challenges. 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

$$\mathbf{z}^{(j)} = \sum_i c_i^A P_{j,i} \mathbf{s}_i. \tag{258}$$

The selectors retain 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. We retain the canonical intervals $\mathcal{I}_j$ for the opening and inner image digits: source holders send their short raw image contributions to the designated image owner, who adds them before digit decomposition. Thus the same witness order and aggregate verifier equations apply. Each machine computes and tests its own response under the common nonce; grinding exchanges only the success indicators. Commitment and quotient computation aggregate short linear images, while the range-check factors are pointwise functions of the locally held witness. The sum-check below therefore starts with its private factors already colocated.

We need only align ownership boundaries to ring elements, which moves at most $O(Jd_{A,\mathrm{next}})$ base-field coefficients for J machines. If we also want balanced intervals, the large response segments cancel from the balancing cost. Write each unit length as $Z + ab_j$, where Z is the common response length, a is the auxiliary length per source block, and the block counts b_j differ by at most one. With a shared tail of length H, balancing moves $O(J(a + H + d_{A,\mathrm{next}}))$ coefficients; for multiple groups, replace a by the sum of their auxiliary widths. This is $\widetilde{O}(1)$ in the source length when the worker count and these auxiliary parameters are polylogarithmic. We can instead retain the unequal intervals whenever their local response bounds fit the schedule, as quantified in Proposition 11.1. This transfer concerns consecutive folds that retain the same workers; the eventual consolidation onto fewer machines has its own communication cost.

11.5 Distributed Sum-Check and Verifier Work

The range and relation checks require more than adding local images: each worker must hold all factors of a product before evaluating it. We distribute the sum-check terms with this requirement in mind 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 and the explicit $\mathbf{x}$ argument accounts for public factors. Let each polynomial have $L = 2^\ell$ Boolean evaluations, and let D bound the degree of a round

polynomial. At the start of each round, every remaining Boolean index has one owner, which holds the evaluations of all t polynomials at that index, with previous challenges substituted. Each owner holds a consecutive interval of indices. 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.

We eliminate the least significant remaining index bit first. For each Boolean suffix $\mathbf{v}$, assign the pair $(0,\mathbf{v}),(1,\mathbf{v})$ to the owner of its even endpoint. If the odd endpoint has another owner, that owner sends its t current partial evaluations to the pair owner. The pair owner now has 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. Each pair owner stores $f_a(r,\mathbf{v})$ as the new entry at index $\mathbf{v}$. Thus the same ownership condition holds in the next round, including 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 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 any round, at most $J-1$ pairs cross interval boundaries. Exchanging their factor values costs at most $t(J-1)$ field elements per round. Workers also send at most $J(D+1)$ round coefficients, and the coordinator broadcasts the challenge. Across ℓ rounds, communication is therefore $O(J(t{+}D)\ell)$ field elements, in addition to the initial redistribution and final factor openings. These messages accompany the short commitment images and scalar claims; response chunking avoids aggregating the full response vectors across machines.

The verifier's setup scan also remains shared. All response chunks use the same public matrices $\mathbf{A},\mathbf{B},\mathbf{D}$. The N_{chunk} response segments contribute separate weights to the $\mathbf{A}$ columns, which the verifier adds before reading the setup:

$$\Omega_A^{\mathsf{chunk}}(a) := \sum_{j\in[N_{\mathsf{chunk}}]} \Omega_A^{(j)}(a). \tag{259}$$

It then evaluates each setup coefficient once. The opening and inner-image segments already partition the ordinary $\mathbf{D}$ and $\mathbf{B}$ inputs, so their setup contributions are partitioned rather than replicated.

The extra verifier work is constructing and adding the N_{chunk} response-weight fragments. We next account for the larger recursive witness and the conditions needed for completeness and security.

11.6 Cost, Completeness, and Security

Response chunking is not an optimization of an isolated Akita proof. The prover forms and decomposes N_{chunk} full-width responses, commits to the larger witness of (246), and runs the remaining recursion over that witness. The proof may acquire additional sum-check rounds or another fold. We pay this cost only when it avoids response-sized communication or permits a source witness that does not fit on one machine.

The schedule authenticates both N_{chunk} and the number of leading folds that retain chunks. The limit $N_{\mathsf{chunk}} \le 64$ bounds the fragmentation an untrusted prover can impose on the verifier.

For completeness, a nonce must work for all chunks at once. We therefore apply the response bound using the maximum number C_j of source blocks contributing to any one response position of chunk j, then take a union bound over chunks. As in the ordinary fold, we condition on the honest transcript prefix before the nonce search.

Proposition 11.1 (Completeness of a response-chunked fold). *Fix a reachable honest transcript prefix and condition on it. For the balanced-digit response model, let C_j bound the number of source blocks contributing to any one response position of chunk j, let σ_∞ bound the absolute values of source coefficients,*

and set $J_+ := \{j \in [N_{\mathsf{chunk}}] : C_j > 0\}$. *Empty source intervals have an identically zero honest response. For each* $j \in J_+$, *let*

$$V_j := C_j \kappa_2^2 \sigma_\infty^2 \tag{260}$$

be the uniform coordinate-variance bound from Lemma G.1. Whole-block ownership gives $C_j = |\mathcal{I}_j|$. *For* (258), *an interval of* n_j *source ring positions gives* $C_j = \lceil n_j/M \rceil$; *balanced ring intervals therefore 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* (108), *then the one-shot failure probability is at most*

$$\delta_{\mathsf{chunk}} := \Pr\left[\exists j \in J_+ : \|\mathbf{z}^{(j)}\|_\infty > t_j \;\middle|\; \boldsymbol{c}_h \neq \perp\right] \le \sum_{j \in J_+} 2n_z \exp\left(-\frac{t_j^2}{2V_j}\right). \tag{261}$$

The one-hot path applies its corresponding chunk-local response bound to the same source-position subsets. The bound is conditional on successful challenge construction and, for a filtered family, requires its accepted distribution to satisfy the premises of the invoked dense or one-hot tail. The schedule chooses the digit depth and retry limit against the complete right-hand side, because accepting a nonce requires all chunks to succeed simultaneously. If $\underline{a}_h$ *is the certified challenge-construction success bound and the verifier admits* N_{try} *fresh nonce trials, the probability that an honest prover exhausts them is at most*

$$(1 - \underline{a}_h(1 - \delta_{\mathsf{chunk}}))^{N_{\mathsf{try}}}. \tag{262}$$

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 retains at most C_j independently challenged source ring elements. Apply the coefficient tail bound of (321) with this count, then take a union bound over its n_z base-field coefficients. A second union bound over J_+ gives (261); it does not require the chunk responses to be independent. The bounded challenge-sampler lemma and distinct random-oracle nonces give (262) and the expectation bound.

We next return to what the aggregate relation proves. Its response equations authenticate the effective response

$$\mathbf{z}_\Sigma := \sum_{j \in [N_{\mathsf{chunk}}]} \mathbf{z}^{(j)}. \tag{263}$$

When the extractor compares two transcript branches, several retained chunks may differ, even though only one folding challenge was changed. The range check bounds the difference in each chunk separately. Suppose that for chunk j this coefficient-wise bound is $\Delta^{\mathsf{cert}}_{\mathsf{rsp},j}$. To bound the difference in $\mathbf{z}_\Sigma$, we add these certified bounds:

$$\Delta^{\mathsf{cert}}_{\Sigma} := \sum_{j \in [N_{\mathsf{chunk}}]} \Delta^{\mathsf{cert}}_{\mathsf{rsp},j}. \tag{264}$$

When all chunks use the same certified alphabet and digit depth, this becomes $\Delta^{\mathsf{cert}}_{\Sigma} = N_{\mathsf{chunk}} \Delta^{\mathsf{cert}}_{\mathsf{rsp}}$. This aggregate bound is the response-difference parameter used by the $\mathbf{A}$ binding argument.

Theorem 11.2 (Extraction of a response-chunked fold). *Fix a nonterminal response-chunked fold whose relation is* (253), *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*

$$\bar{\beta}_\infty = \Delta^{\mathsf{cert}}_{\Sigma}. \tag{265}$$

Consequently, if $\bar{\kappa}_1$ *bounds the extracted challenge difference, the coefficient-norm* $\mathbf{A}$ *collision of Lemma 10.23 has radius at most*

$$\eta_{A,\mathsf{chunk}} = 2\bar{\kappa}_1 \Delta^{\mathsf{cert}}_{\Sigma}. \tag{266}$$

For a raw challenge family drawn from an ℓ_1*-ball of mass* ω, *one may take* $\bar{\kappa}_1 \le 2\omega$, *and hence*

$$\eta_{A,\mathsf{chunk}} = 4\omega \Delta^{\mathsf{cert}}_{\Sigma}. \tag{267}$$

All $\mathbf{B}$, $\mathbf{D}$, $\mathbf{F}$, *and* $\mathbf{H}$ *collision radii remain those of the ordinary fold.*

Proof. Apply the strong extractors below each folding-challenge child. They recover the exact aggregate identities (248) through (252), 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. Condition on all such digits being 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 from Theorem 10.31 to the horizontally concatenated relation gives

$$\iota(\bar{c}_q)\mathbf{s}_q = \sum_{j\in[N_{\mathsf{chunk}}]} \left(\mathbf{z}_j^{(q,1)} - \mathbf{z}_j^{(q,0)}\right). \tag{268}$$

The certified range of chunk j bounds its summand by $\Delta^{\mathsf{cert}}_{\mathsf{rsp},j}$ coefficient-wise. The triangle inequality therefore gives (265). This is the point at which the factor N_{chunk} appears: the aggregate rows do not identify which response chunk changed between two transcript 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 (266); the challenge-shell specialization gives (267). Chunking neither duplicates the other semantic row families nor changes their certified alphabets, so their collision radii are unchanged.

We can now apply the schedule-level guarantees 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 10.44, 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 (266), or equivalently (267) for the scheduled challenge shell;*
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 10.44 hold without any additional error term. In particular, the advantage bound remains (233), 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. In the backward induction of Theorem 10.44, replace Theorem 10.31 at a chunked level by Theorem 11.2. The rest of the induction uses 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 (266). This gives the stated substitutions in (233).

This security statement treats all machines and the aggregator jointly as one prover for the chunked relation. With the aggregate coefficient envelope charged 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 give us a family of admissible Akita schedules. We now ask how to choose 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 [67]. LaZer's newer LaBRADOR implementation instead generates a sequence offline through successive local size improvements [69]; the inspected RoKoKo artifact supplies concrete chains, with automatic selection listed as future work [87]. 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 (41); 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 (106). 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 (61). Shared setup storage is the largest required prefix, not the sum of the prefix lengths. These dependencies are included before a candidate is scored.

Comparing costs. We record proof length, verifier work, prover work, setup storage, and distributed communication as

$$\mathsf{cost}(\mathsf{sch}) = (C_{\mathsf{pf}}, C_{\mathsf{V}}, C_{\mathsf{P}}, C_{\mathsf{setup}}, C_{\mathsf{comm}}). \tag{269}$$

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 6; 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

$$\underset{\substack{\mathsf{sch}\text{ in the family specified by }\mathcal{P}\\ \mathsf{sch}\text{ admissible and within resource limits}}}{\text{minimize}} \quad \mathsf{Score}_{\mathcal{P}}(\mathsf{sch}). \tag{270}$$

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 G.1 and Theorem G.3 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 G.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 (184): 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 G.9 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 G.7 and the digit squared-norm bound of (362), 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 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 (270). 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 [64,65].

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 C, the schedule-wide bounds of Theorem 10.44, 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 10.3. Neither validation nor proof verification requires repeating the search or establishing optimality.

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 [65,11]. 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 [64].

The implementation and benchmark artifacts are publicly available [65,68]. Speedup ratios use unrounded timing medians.

Workloads and metrics. Each standalone experiment measures a native commitment-and-opening workload; Appendix H 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 their communication cost 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 retain their native polynomial interfaces, so equal input sizes do not imply identical statements.

Equal nominal payloads 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 H details these workload differences.

PCS Benchmark Software. All implementations used in our polynomial commitment schemes comparison are collected in [68]. We imported implementations of the hash-based and lattice-based schemes and integrated them into a common benchmarking framework.

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 uses structured vSIS priced using GSA/MATZOV-classical, and a stated 100-bit statistical soundness target [61, Section 9.3]. Its approximate low-norm challenge sampling has an estimated noninvertibility probability of about 2^{-94} [61, Section 9.1]. Appendix F.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 F 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 8 gives all five measured input sizes; Figure 10 shows the full scaling curves.

Calibrating Greyhound. To align the concrete security targets, we make three changes to Greyhound's reference implementation (Appendix I). 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.

Table 8: Lattice PCS measurements across the full input-size range. Proof size excludes the commitment. RoKoKo uses its native instances as described above; a dash denotes an unsupported input. The full size range appears in Figure 10.

Bits committed	Scheme	Commit (s)	Open (s)	Verify (ms)	Proof (KB)	Peak RAM (GiB)
2^{27}	Akita direct	0.039	0.510	6.1	57.7	0.108
	Akita offloaded	—	—	—	—	—
	Greyhound	0.111	0.187	77.4	60.1	0.33
	RoKoKo	0.038	0.144	4.1	109.3	0.85
2^{29}	Akita direct	0.141	0.704	7.2	58.2	0.223
	Akita offloaded	0.139	1.07	6.7	62.6	0.232
	Greyhound	0.442	0.393	148	60.6	1.08
	RoKoKo	0.129	0.246	3.9	109.4	1.7
2^{31}	Akita direct	0.536	1.35	11.5	59.5	0.627
	Akita offloaded	0.534	1.86	9.9	62.7	0.671
	Greyhound	2.10	1.04	329	63.3	3.93
	RoKoKo	0.471	0.668	4.6	114.8	4.47
2^{33}	Akita direct	2.07	2.32	18.1	60.9	1.21
	Akita offloaded	2.22	3.34	11.4	63.3	1.33
	Greyhound	11.6	4.60	715	64.9	20.7
	RoKoKo	1.83	1.63	4.3	114.8	12.3
2^{35}	Akita direct	8.42	5.69	32.2	61.0	4.37
	Akita offloaded	10.7	8.35	14.1	63.7	4.47
	Greyhound	54.1	20.1	1537	67.2	92.6
	RoKoKo	8.90	4.70	7.2	115.0	36.9

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 8).

The direct/offloaded comparison shows what setup offloading buys. The first supported offloaded standalone instance commits 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 increasing opening time 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 first level 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 8). Most of the fp32 prover increase at 2^{35} bits comes from that first level and not from opening the setup: the offloaded root uses five source digits instead of four and heavier fold challenges, as Section 2.4 explains. At the same size, offloading changes fp64 opening time from 4.90 to 5.46 seconds and fp128 opening time from 6.35 to 6.21 seconds.

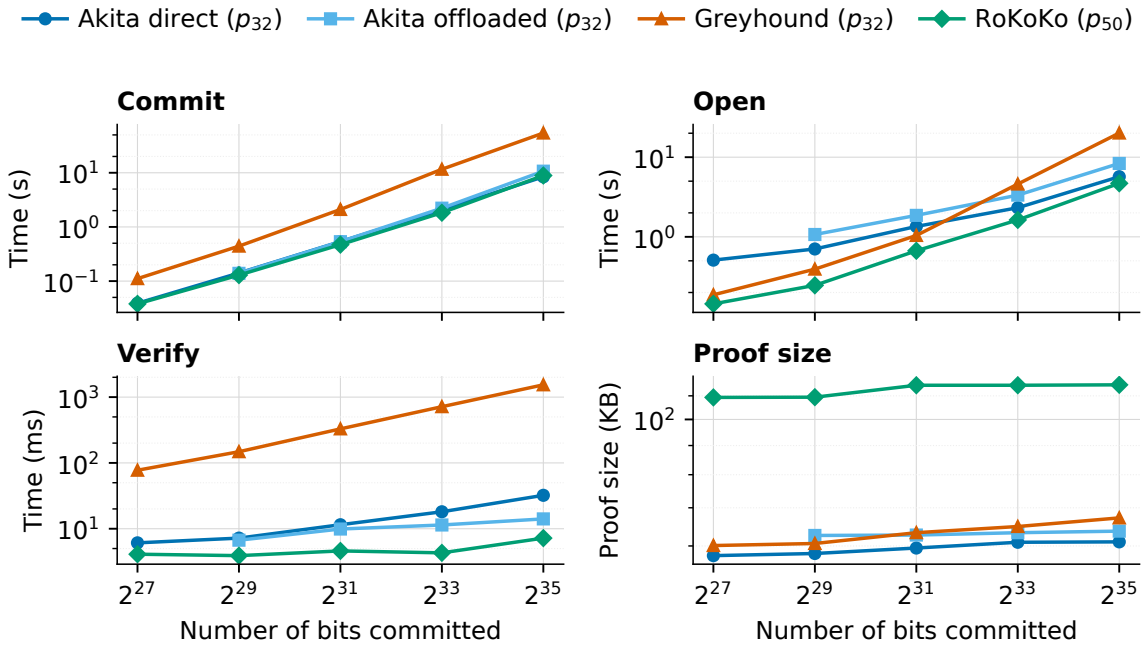

Fig. 10: Lattice PCS comparison. Input size is the nominal number of bits committed, $B = N \log_2 |\mathbb{F}|$; RoKoKo uses its closest supported native instance. Proof size excludes the commitment and evaluation. Missing points denote unsupported inputs.

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 8).

13.3 Comparison with Hash-Based PCSs

We compare Akita with FRI [16], STIR [8], WHIR [9], BaseFold [92], and Ligerito [84] 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 H 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 H).

Figure 11 shows 2^{35} bits committed, where setup offloading materially reduces Akita's verifier time. Table 9 gives eight-thread prover measurements at 2^{27}, 2^{31}, and 2^{35} bits; the graph shows both prover thread counts at 2^{35}.

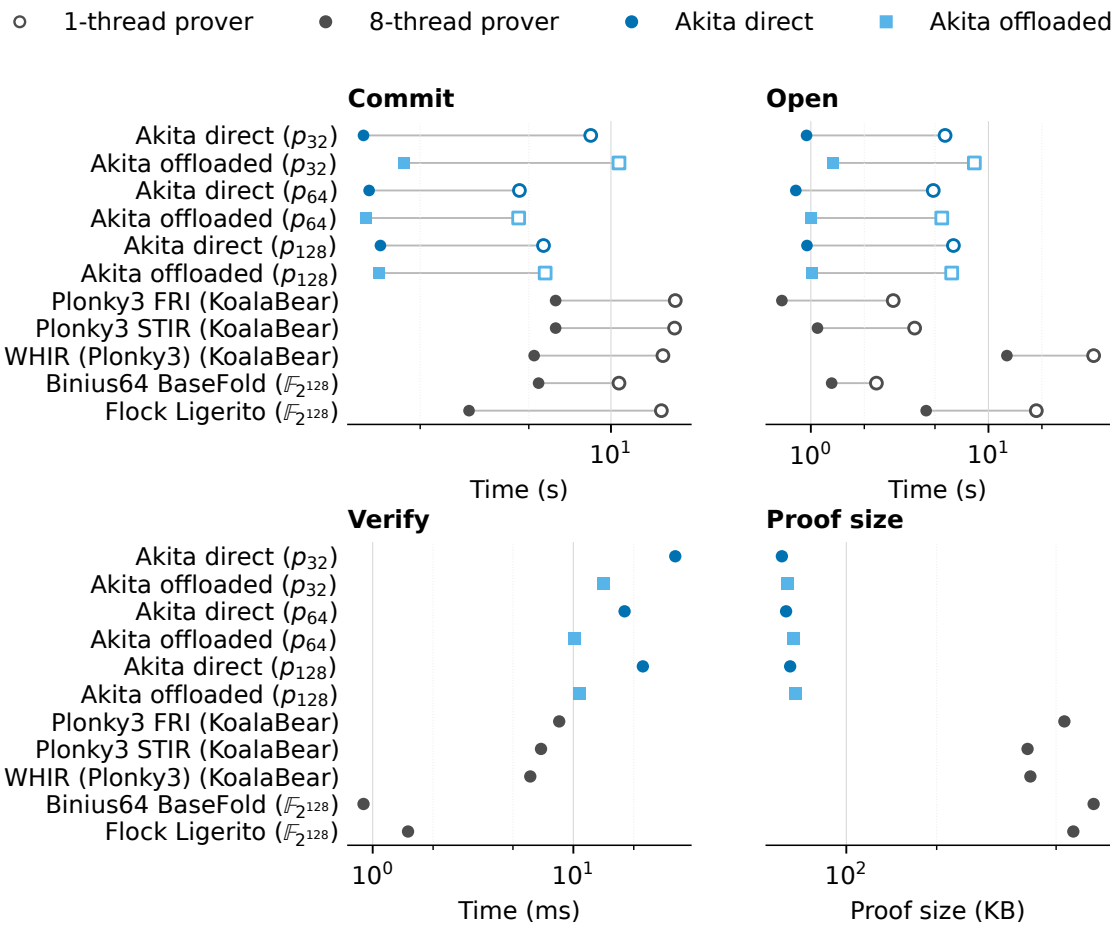

Fig. 11: Hash PCS comparison at 2^{35} bits committed. Commitment and opening show one- and eight-thread proving. Proof size excludes the commitment. Plonky3 FRI and STIR use batched univariate instances. Baselines without a completed measurement at this size are omitted.

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 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 11 and Table 9). These are workload-specific tradeoffs: the field and packing choices change the opening problem as well as the implementation cost.

Table 9: Hash PCS measurements at three input sizes. Commitment, opening, and peak RAM use eight prover threads. Akita field subscripts denote prime-field bit widths. Rows marked $^{(1)}$ use batched univariate FRI/STIR, which these libraries need only at the two larger sizes (Appendix H). OOM denotes the memory limit; n/m marks a size that was not measured.

Bits committed	Scheme	Commit (s)	Open (s)	Verify (ms)	Proof (KB)	Peak RAM (GiB)
2^{27}	Akita (p_{32})	0.00827	0.143	6.0	57.7	0.147
	Akita (p_{64})	0.00817	0.0966	3.9	59.4	0.133
	Akita (p_{128})	0.00977	0.0789	4.2	61.7	0.153
	Plonky2 FRI	0.548	1.2	14.5	637.2	0.957
	Plonky3 FRI	0.205	0.241	7.1	410.6	1.17
	Plonky3 STIR	0.21	0.43	5.8	290.9	1.17
	WHIR (Plonky3)	0.0205	0.146	3.4	211.4	0.24
	Binius64 BaseFold	0.00664	0.00725	0.4	308.6	0.0913
	Flock Ligerito	0.0094	0.0185	1.2	408.9	0.0905
	WHIR (WorldFnd)	0.0673	0.348	1.0	196.6	0.181
	BaseFold (SP1)	2.17	3.25	25.0	1179.4	1.33
2^{31}	Akita (p_{32})	0.0748	0.275	11.7	59.5	0.663
	Akita offload (p_{32})	0.0736	0.364	10.2	62.7	0.696
	Akita (p_{64})	0.0499	0.191	7.3	62.4	0.454
	Akita offload (p_{64})	0.0463	0.24	5.9	64.7	0.519
	Akita (p_{128})	0.0742	0.188	9.1	64.3	0.507
	Plonky2 FRI	9.5	20.3	19.5	846.0	15.3
	Plonky3 FRI$^{(1)}$	0.686	0.46	7.8	452.7	2.78
	Plonky3 STIR$^{(1)}$	0.687	0.857	6.1	312.4	2.78
	WHIR (Plonky3)	0.445	3.04	4.4	279.9	3.7
	Binius64 BaseFold	0.23	0.0923	0.6	472.8	1.42
	Flock Ligerito	0.165	0.265	1.2	494.6	1.35
	WHIR (WorldFnd)	1.37	6.3	1.2	256.9	2.85
	BaseFold (SP1)	5.39	3.32	25.4	1194.8	2.5
2^{35}	Akita (p_{32})	1.24	0.946	32.2	61.0	4.4
	Akita offload (p_{32})	1.75	1.33	14.1	63.7	4.5
	Akita (p_{64})	1.3	0.823	18.0	63.0	4.61
	Akita offload (p_{64})	1.27	0.997	10.1	66.8	4.77
	Akita (p_{128})	1.43	0.951	22.2	65.0	4.81
	Akita offload (p_{128})	1.41	1.01	10.7	67.9	4.98
	Plonky2 FRI	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*
	Plonky3 FRI$^{(1)}$	6.27	0.687	8.5	532.7	12.2
	Plonky3 STIR$^{(1)}$	6.27	1.09	6.9	401.6	12.2
	WHIR (Plonky3)	5.23	12.7	6.1	410.4	18.9
	Binius64 BaseFold	5.43	1.31	0.9	666.7	22.7
	Flock Ligerito	3.02	4.46	1.5	570.5	21.6
	WHIR (WorldFnd)	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*
	BaseFold (SP1)	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*

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 10 gives trace lengths $2^{17}, 2^{19}, \ldots, 2^{27}$; the graphs retain all thirteen measured lengths.

Table 10: Jolt measurements at every other padded trace length on the M4 Max. Times and peak RAM are medians of three runs. Proof size is the complete Jolt proof, in decimal KB. Akita first uses setup offloading at $T = 2^{21}$; Jolt-with-Dory widens its one-hot chunks at $T = 2^{25}$.

Trace length	Backend	Prove (s)	Verify (ms)	Proof (KB)	Peak RAM (GiB)
2^{17}	Akita	0.33	8.29	81.8	0.30
	Dory	1.16	67.51	81.9	0.24
2^{19}	Akita	0.59	11.50	84.4	0.45
	Dory	2.38	71.04	86.2	0.48
2^{21}	Akita	1.73	13.28	88.7	1.06
	Dory	5.41	75.26	90.6	1.20
2^{23}	Akita	5.55	19.30	90.0	1.97
	Dory	14.78	79.46	94.9	2.98
2^{25}	Akita	20.72	23.33	91.5	6.43
	Dory	54.31	82.16	91.8	9.71
2^{27}	Akita	75.05	24.88	93.6	24.37
	Dory	171.93	86.19	95.9	27.49

Proving and memory. Across these trace lengths, Jolt-with-Akita proves 2.2–4.2× faster than Jolt-with-Dory (Figure 12). 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.

Verification and proof size. Akita reduces verification time by 2.8–8.5× (Figure 13). 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

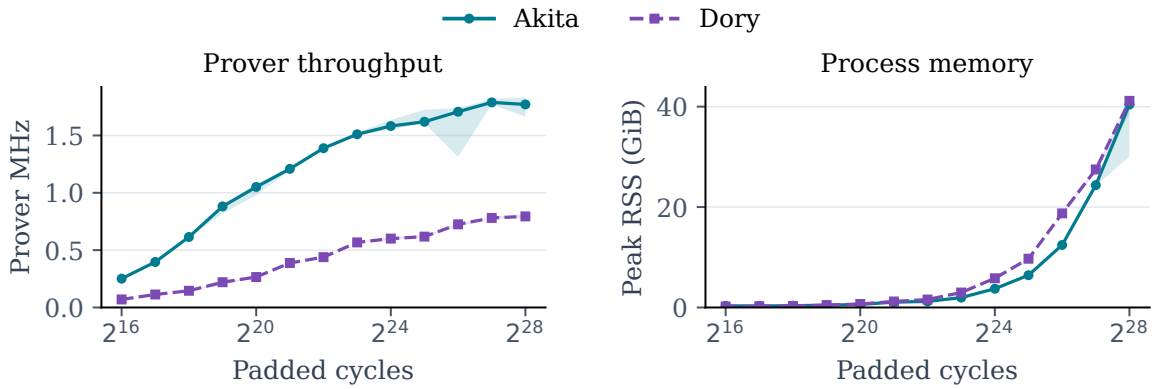

Fig. 12: SHA-256 chains on the M4 Max. Left: padded prover throughput. Right: process peak RSS. Points show medians of three runs; bands show observed ranges.

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 12 and 13).

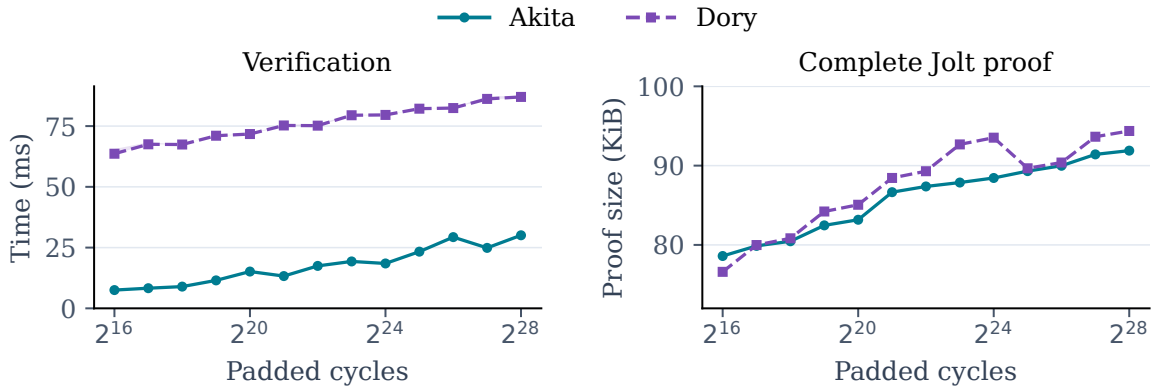

Fig. 13: SHA-256 chains on the M4 Max. Left: verification time, showing medians and observed ranges over three runs. Right: complete proof size in KiB.

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, with many more promising avenues for impact. In particular, Akita facilitates the development of better proof systems for verifying natively

lattice-based claims, such as aggregating the post-quantum lattice signatures ML-DSA [78] and Falcon (FN-DSA) [43]. 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 [91].

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, and GPT 6 Astra 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. 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
34. 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

35. 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
36. 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
37. 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
38. 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
39. Ethereum Foundation: Ethproofs: Tracking real-time proving for ethereum. https://ethproofs.org/ (2025)
40. Ethereum Foundation: Shipping an L1 zkevm #1: Realtime proving. https://blog.ethereum.org/2025/07/10/realtime-proving (2025)
41. Ethereum Foundation: The proximity prize. https://proximityprize.org/ (2026)
42. 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
43. 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
44. Gabizon, A., Williamson, Z.J., Ciobotaru, O.: PLONK: Permutations over Lagrange-bases for oecumenical non-interactive arguments of knowledge. ePrint Report 2019/953 (2019)
45. 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
46. Goldwasser, S., Kalai, Y.T., Rothblum, G.N.: Delegating computation: interactive proofs for muggles. Journal of the ACM (JACM) **62**(4), 1–64 (2015)
47. Golovnev, A., Lee, J., Setty, S., Thaler, J., Wahby, R.S.: Brakedown: Linear-time and post-quantum snarks for R1CS. Cryptology ePrint Archive (2021)
48. Groth, J.: On the size of pairing-based non-interactive arguments. In: Advances in Cryptology – EUROCRYPT 2016 (2016)
49. Gruen, A.: Some improvements for the PIOP for ZeroCheck. Cryptology ePrint Archive, Paper 2024/108 (2024), https://eprint.iacr.org/2024/108
50. 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
51. 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
52. 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
53. 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
54. 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
55. 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
56. 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
57. 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
58. 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)
59. 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
60. 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

61. 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
62. 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
63. 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
64. LayerZero Labs: The akita book. https://github.com/LayerZero-Labs/akita/tree/main/book (2026)
65. LayerZero Labs: Akita codebase. https://github.com/LayerZero-Labs/akita (2026)
66. LayerZero Labs: Akita v0.1.0. https://github.com/LayerZero-Labs/akita/tree/029ceceaba879a0d9bf80a47d2e08534e630457d (2026), release revision inspected September 11, 2026
67. 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
68. LayerZero Labs: PCS Benchmarks. https://github.com/LayerZero-Labs/pcs-benchmarks (2026)
69. 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
70. leanEthereum: leanVM: Minimal hash-based zkvm for a post-quantum ethereum. https://github.com/leanEthereum/leanVM (2026)
71. Lee, J.: Dory: Efficient, transparent arguments for generalised inner products and polynomial commitments. In: Theory of Cryptography Conference. pp. 1–34. Springer (2021)
72. 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
73. Lund, C., Fortnow, L., Karloff, H., Nisan, N.: Algebraic methods for interactive proof systems. In: FOCS (Oct 1990)
74. 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
75. 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
76. 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
77. 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
78. 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
79. 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
80. Nguyen, N.K., O'Rourke, G., Zhang, J.: Personal communication concerning the base-field optimization in Hachi (2026)
81. 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
82. Nguyen, W., Setty, S.: Neo: Lattice-based folding scheme for ccs over small fields and pay-per-bit commitments. Cryptology ePrint Archive (2025)
83. 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
84. 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

85. Osadnik, M.: PikkuFold: Efficient folding in a few kilobytes. Cryptology ePrint Archive, Paper 2026/1809 (2026), https://eprint.iacr.org/2026/1809
86. 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/
87. 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
88. 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)
89. 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
90. Succinct: Sp1 hypercube: Proving ethereum in real-time. https://blog.succinct.xyz/sp1-hypercube/ (2025)
91. 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
92. 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

A Multilinear Evaluation from Structured Factors

To obtain a succinct verifier, we must do more than reduce the fold checks to a few polynomial evaluations. The sum-checks leave public weight polynomials that the verifier must evaluate for itself. If we enumerate their Boolean evaluations over the entire witness or setup domain, this last step can cost as much as the work we sought to avoid. We therefore need evaluation algorithms that use the structure of these weights directly.

We isolate the general evaluation problem here. The specialization to Akita's opening, range, norm, and setup weights is given in Appendix B. That appendix also develops the residue-kernel calculations needed for quotient-free ring checking.

The public weights have simple factors in their logical coordinates: a block index selects a folding challenge, a digit index selects a gadget power, or an opening coordinate selects an equality weight. The difficulty is where these entries occur in the committed vector. Offsets, tight strides, and interleaved coordinates need not respect the bit boundaries of those factors, so the usual tensor factorization of an equality polynomial does not apply directly.

We resolve this mismatch by retaining only the information that one part of the address calculation passes to the next. For an offset, that information is an addition carry; a truncated interval also requires a boundary condition. We first express the desired evaluation as an inner product with an embedded tensor, then give a finite-state calculation of that inner product. Finally, we relate this calculation to weighted read-once branching programs and specialize it to affine addresses. This separates the common evaluation argument from the particular weights and layouts in the main body.

A.1 From Tensor Factors to Vector Addresses

Let the logical index be split into blocks $x = (x_1, \dots, x_k)$, with $x_t \in \mathcal{X}_t$, and suppose its weight is $u(x) = \prod_{t=1}^{k} u_t(x_t)$. An address map A places each logical entry at an n-bit vector address. We identify addresses with little-endian bit vectors throughout this appendix. The evaluation we seek is

$$T_A(\mathbf{r}; u) := \sum_{x \in \mathcal{X}_1 \times \cdots \times \mathcal{X}_k} \left(\prod_{t=1}^{k} u_t(x_t) \right) \widetilde{\mathsf{eq}}(\mathbf{r}, A(x)). \tag{271}$$

To see its multilinear meaning, place the logical weights into a vector indexed by Boolean addresses by setting

$$U_A(y) := \sum_{x : A(x) = y} u(x).$$

Then $T_A(\mathbf{r}; u) = \widetilde{U}_A(\mathbf{r})$. This definition also handles overlapping contributions: if several logical entries have the same address, their weights add. Enumerating every logical entry takes work proportional to $\prod_t |\mathcal{X}_t|$. We want to use the separate factor values whenever the address map allows it.

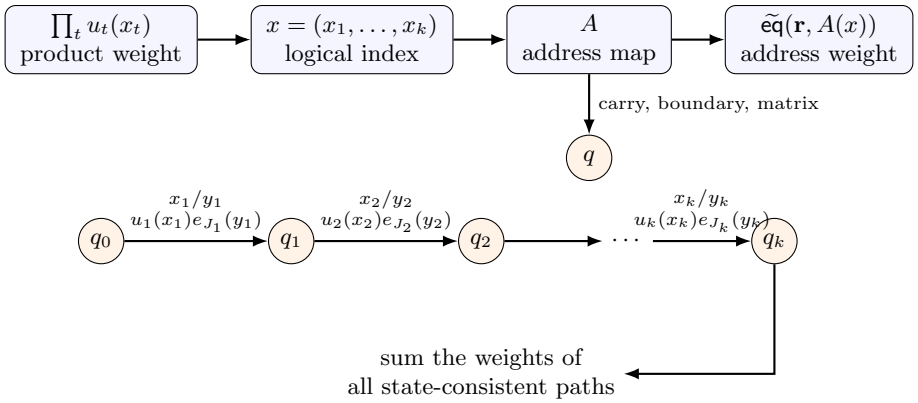

Fig. 14: The logical tensor factors do not need to align with the address coordinates of the equality polynomial. The state q carries exactly the information by which earlier blocks affect later address bits, and $e_J(y) = \widetilde{\mathsf{eq}}(\mathbf{r}_J, y)$. The lower row is both a dynamic program and a layered weighted branching program.

Aligned coordinates. Suppose $A(x)$ is the concatenation of blocks $A_t(x_t)$ occupying disjoint sets J_t of address bits. The equality weight then separates, giving

$$T_A(\mathbf{r}; u) = \prod_{t=1}^{k} \left(\sum_{x_t \in \mathcal{X}_t} u_t(x_t) \widetilde{\mathsf{eq}}(\mathbf{r}_{J_t}, A_t(x_t)) \right).$$

We evaluate each factor separately and multiply the results. Once the local equality weights are available, the contraction takes work proportional to $\sum_t |\mathcal{X}_t|$. There is no reason to enumerate all entries of the tensor product. We next extend this evaluation method to shifted entries and tighter packing.

A shift couples the factors through a carry. Write $x = q + 4h$, with $q \in [4]$, and take $A(q, h) = 3 + q + 4h$. For simplicity, assume all these addresses fit the vector domain. If $3 + q = y + 4c$, then $y \in [4]$ and $c \in \{0, 1\}$. We can therefore summarize the low factor in two scalars:

$$L_c := \sum_{\substack{q \in [4] \\ \lfloor (3+q)/4 \rfloor = c}} u_{\mathsf{lo}}(q) \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{lo}}, (3 + q) \bmod 4).$$

The high factor needs to know only which carry occurred. Consequently,

$$T_A(\mathbf{r}; u) = \sum_{c=0}^{1} L_c \sum_{h} u_{\mathsf{hi}}(h) \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{hi}}, h + c).$$

The offset has replaced one product by a sum of two products. We retain the small piece of information passed between the two blocks, rather than the entire low index. The finite-state calculation below makes this observation precise for several carries and boundary conditions.

A.2 The Common Evaluation Problem

Some verifier weights depend on 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. Both positions contribute equality factors. We therefore allow several address maps, each with its own product of bit weights.

For $a \in [p]$, let $A_a(x)$ be an n_a-bit address and define

$$B_a(y) := \prod_{j=0}^{n_a - 1} b_{a,j}(y_j), \qquad b_{a,j} : \{0, 1\} \to \mathbb{F}.$$

Equality at $\mathbf{r}_a$ is the choice $b_{a,j}(0) = 1 - r_{a,j}$ and $b_{a,j}(1) = r_{a,j}$. The choice $b_{a,j}(0) = 1$, $b_{a,j}(1) = z_{a,j}$ gives monomial weights. A geometric sequence α^y is another example: take $b_{a,j}(0) = 1$ and $b_{a,j}(1) = \alpha^{2^j}$. These choices use the same address calculation.

Let $D \subseteq \mathcal{X}_1 \times \cdots \times \mathcal{X}_k$ be the logical domain included in the sum. Our common task is to compute

$$T := \sum_{x \in D} \left(\prod_{t=1}^{k} u_t(x_t) \right) \prod_{a \in [p]} B_a(A_a(x)). \tag{272}$$

All addresses in this expression are in their declared domains. When a formula is used outside a declared address domain, we assign it weight zero; we never interpret address overflow modulo the domain size. The included intervals and omitted coordinates are specified by D.

The paired-equality case will recur in the applications:

$$\sum_{x \in D} u(x) \widetilde{\mathsf{eq}}(\mathbf{r}_L, A_L(x)) \widetilde{\mathsf{eq}}(\mathbf{r}_R, A_R(x)). \tag{273}$$

It is an instance of the single-address formulation by concatenating A_L and A_R, but retaining the two maps makes the separate carries easier to follow. More generally, we can add expressions of (272) with different domains, weights, and offsets. This is how we represent a union of witness intervals or several matrix views contributing to the same setup coefficient.

We distinguish two ways to specify a factor u_t. A dense factor is supplied by its values, which must be read. A factor on a Boolean block can instead be supplied as a product over its bits. For example, $u_t(x_t) = \prod_j w_{t,j}(x_{t,j})$ has a description linear in the number of bits, even though its vector of values is exponentially larger. We retain that description when evaluating it. The cost depends on this representation, as well as on the address maps.

A.3 Finite-State Contraction

We now describe the information passed between logical blocks. Output bits may belong to any of the address maps in (272). For notation, concatenate all their coordinates into one set J and write b_j for the corresponding bit weight. Partition J into sets $J_1, \ldots, J_k$, allowing an empty set when a layer reads input without yet outputting any address bits. A singleton input alphabet of weight one allows a final layer to output remaining carry bits without reading another logical coordinate.

Definition A.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*

$$\Gamma_t \subseteq Q \times \mathcal{X}_t \times Q \times \{0,1\}^{J_t} \times \mathbb{F}.$$

A transition $(q, x_t, q', y_t, \eta_t)$ reads x_t, outputs y_t, moves from q to q', and contributes the scalar η_t. The transducer assigns to a logical input the sum of its path weights:

$$v(x) := \sum_{\pi : x} \omega_{\mathsf{fin}}(q_k) \prod_{t=1}^{k} \left(\eta_t \prod_{j \in J_t} b_j(y_{t,j}) \right). \tag{274}$$

It is deterministic if there is at most one outgoing transition for each pair (q, x_t). In that case, write the partial transition function as $\tau_t(q, x_t) = (q', y_t, \eta_t)$.

For a deterministic address calculation, take the transition scalars to be one, output the address bits, and accept exactly the inputs in D. The resulting $v(x)$ is $\mathbf{1}_D(x) \prod_a B_a(A_a(x))$. An undefined transition or a zero final weight rejects an input. Several weighted paths also allow logical contributions to add; their addresses need not have disjoint images.

The dynamic program keeps one accumulated field element per state. Initialize $z_0(q_{\mathsf{in}}) = 1$ and all other entries to zero. At layer t, start a zero next-state vector and, for each transition, add

$$z_t(q') \mathrel{+}= z_{t-1}(q) u_t(x_t) \eta_t \prod_{j \in J_t} b_j(y_{t,j}). \tag{275}$$

Paths that reach the same state have the same possible continuations, so we add their weights before processing the next block.

Theorem A.2 (Finite-state structured inner product). *The recurrence (275) returns*

$$\sum_{q\in Q} z_k(q)\omega_{\mathsf{fin}}(q) = \langle u_1 \otimes \cdots \otimes u_k, v\rangle\,.$$

Let $G_t = |\Gamma_t|$, $h_t = |J_t|$, and $G_{\mathsf{fin}} = |\operatorname{supp}(\omega_{\mathsf{fin}})|$. With the local factor and bit weights available, direct contraction uses $O(\sum_t G_t(h_t+1)+G_{\mathsf{fin}})$ field operations and $O(|Q|)$ field elements for the current and next state vectors. More precisely, successive multiplication along each transition, with no shortcuts for zero or unit factors, uses

$$\sum_t G_t(h_t+2)+G_{\mathsf{fin}} \quad \textit{multiplications, and} \quad \sum_t G_t + \max(G_{\mathsf{fin}}-1,0) \quad \textit{additions.}$$

These counts exclude initialization, construction of the transitions, and preparation of the local weights. If the output-bit products are prepared in time P and stored for lookup, the field-operation bound becomes $O(P+\sum_t G_t + G_{\mathsf{fin}})$; their storage is additional to the state vectors.

Proof. After layer t, $z_t(q)$ is the sum of the weights of all labeled path prefixes ending at q. Each prefix weight contains its logical factors, transition scalars, and output-bit weights. This holds initially and is preserved by (275). Multiplication by the final weights and summation over states therefore gives (274), summed against $u(x)$. Each transition multiplies by its logical factor, its transition scalar, and its h_t bit weights, then adds to the next-state accumulator. The final dot product gives the remaining operations.

The theorem is a bound in the size of the state description, not a claim that every address map has a small description. For deterministic maps, $G_t \le |Q|\,|\mathcal{X}_t|$, so small state gives the desired dependence on the sum of factor sizes. Several carries require their joint state, and arbitrary dense factors cannot be replaced by independent bit weights. We account for these costs explicitly in the affine specializations below. Stored equality weights can be reused for further contractions at the same evaluation point; changing that point requires preparing them again.

Relation to prior evaluation algorithms. The recurrence in Theorem A.2 is the standard evaluation algorithm for a weighted read-once branching program. Each logical symbol selects a sparse state transition, and distributivity contracts its tensor 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 [52]; the theorem above allows arbitrary local factor values rather than only the Lagrange weights of one MLE point.

Jagged Polynomial Commitments applies the same calculation to a width-4 program that tracks one addition carry and one comparison bit [51], the closest cryptographic precedent for Akita's carry states. The transfer-matrix view is also the classical equivalence between weighted automata and matrix-product representations [35]. Akita's contribution here is to identify its non-aligned witness and setup layouts as small-state address transducers and to account for the exact transition cost of evaluating their public weights.

A.4 Affine Addresses on Intervals

For affine layouts, the state has a concrete arithmetic meaning. We start with consecutive digits, then allow separate strides and several address maps. This order lets us identify exactly which part of the cost comes from the digits and which part comes from the layout.

Consecutive digits. When one value occupies δ consecutive digit cells, its address is

$$A(i,u) = o + \delta i + u, \qquad i \in [L], \quad u \in [\delta]. \tag{276}$$

Here o is the first vector address. Neither L nor δ needs to be a power of two. We assume the addresses of the specified interval fit in $[2^n]$.

Lemma A.3 (Reading consecutive-digit addresses). *The bits of (276) can be read from low to high with a carry having $O(\delta)$ possible values at each layer. The calculation can also enforce $i < L$, and does not divide by an evaluation-point coordinate.*

Proof. Read u first and initialize the carry to $c_0 = o + u$. At bit t of i, write $c_t + \delta i_t = y_t + 2c_{t+1}$, with $y_t \in \{0, 1\}$. The output bits are those of $o + u + \delta i$. Indeed, after t steps,

$$c_t = \left\lfloor \frac{o + u + \delta(i \bmod 2^t)}{2^t} \right\rfloor .$$

As u and the processed bits vary, these values lie in an interval of $O(\delta)$ integers. Once the input bits end, continue outputting the carry bits. To enforce $i < L$, pad i and L to a common bit width and retain the comparison of the processed bits with the corresponding bits of L. A new, more significant unequal bit determines the comparison; otherwise the previous comparison remains. This adds only three possible comparison states.

Contracting a low block before reading the high block. The bit recurrence explains the carry, but an arbitrary factor on a block need not factor over its bits. We can instead sum a whole low block into carry buckets and then scan the high factor once per bucket.

Proposition A.4 (Consecutive-digit evaluation). *Let $L, \delta \geq 1$, $o + \delta L \leq 2^n$, and let $Q = 2^m$ with $m \leq n$. Suppose $\psi(i) = \psi_H(h)\psi_L(q)$ for $i = Qh + q < L$. Then*

$$T = \sum_{i<L} \sum_{u<\delta} \psi(i) g(u) \widetilde{\mathsf{eq}}(\mathbf{r}, o + \delta i + u)$$

can be contracted in $O(\delta(Q + \lceil L/Q \rceil))$ field operations once its local equality weights are available for constant-time lookup. Their preparation time P_{eq} is additional, giving total work $O(P_{\mathsf{eq}} + \delta(Q + \lceil L/Q \rceil))$. The contraction uses $O(\delta)$ carry accumulators, in addition to its stored factor values and any prepared equality weights.

Proof. Write $o = Qa + r_o$ with $r_o \in [Q]$, and split $\mathbf{r}$ into m low coordinates and the remaining high coordinates. For a complete logical row, define

$$L_c := \sum_{\substack{q<Q,\ u<\delta \\ \lfloor (r_o + \delta q + u)/Q \rfloor = c}} \psi_L(q) g(u) \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{lo}}, (r_o + \delta q + u) \bmod Q).$$

There are at most $\delta + 1$ buckets, because $0 \leq c \leq \delta$. The contribution of the complete rows is

$$\sum_{h<\lfloor L/Q \rfloor} \psi_H(h) \sum_{c=0}^{\delta} L_c \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{hi}}, a + \delta h + c).$$

Building the buckets takes $O(\delta Q)$ operations, and the high contraction takes $O(\delta \lfloor L/Q \rfloor)$. If L has a partial final row, form another set of buckets restricted to $q < L \bmod Q$ and contract them with that row's high weight. This adds at most $O(\delta Q)$ operations.

The factor δ in the bound accounts for the carry width; it is not an unspecified constant. Nor is the preparation cost automatically free: direct evaluation of an equality weight costs one operation per bit up to constant factors. We describe ways to share this work below. An arbitrary vector $(\psi(i))_{i<L}$ contains L independent values and requires $\Omega(L)$ input accesses. The saving comes from retaining the given high/low factorization, not from treating arbitrary factor values as cheap.

Separate strides and partial windows. The same calculation applies to

$$A(i, u) = o + s(i - i_0) + tu, \qquad i_0 \leq i < i_0 + L, \quad u \in [d], \tag{277}$$

where s, t are positive integers and $s > t(d - 1)$. The inequality ensures that a digit group fits before the next outer position. For $i = Qh + q$, the outer weight remains $\psi_H(h)\psi_L(q)$, using the global logical index i.

The address o locates the first position i_0 in the interval. This distinction matters when a witness interval begins partway through a logical row.

Separate the partial first row, complete rows, and partial last row, counting a row only once if the whole window lies in it. If the portion of a row included in the interval begins at low coordinate q_0, its first vector address is $b_h = o + s(Qh + q_0 - i_0)$. Reduce b_h modulo Q and summarize the (q, u) pairs included in that portion of the row using the low address $(b_h \bmod Q) + s(q - q_0) + tu$ and the logical weight $\psi_L(q)$. There are at most $s+1$ carry buckets. For complete rows, $q_0 = 0$ and successive starting addresses differ by sQ, so their low summary is shared; only the high address changes. Thus, if R is the number of high rows meeting the window, the contraction costs

$$O\big(dQ + (s+1)R\big)$$

with local equality lookup, plus its preparation cost. Empty windows contribute zero. A large outer stride can therefore increase the state space even when the number of digits is small.

Remark A.5 (Power-of-two special case). If the digit count is a power of two and the starting address is aligned to that digit block, the equality polynomial separates across the digit boundary. An unaligned base introduces the carry across that boundary. Both evaluations refer to the same stored vector; padding or moving its entries is unnecessary.

Several address maps. Let $x_1, \dots, x_k$ be nonnegative integer coordinates and consider

$$A_a(x) = o_a + \sum_{t=1}^{k} s_{a,t} x_t \qquad (a \in [p]),$$

with nonnegative strides. A zero stride allows a logical axis to affect only some of the addresses. When each x_t is given on a Boolean block and its weight factors over bits, read one bit from each active axis at a time. Initialize $c_{a,0} = o_a$ and update by

$$c_{a,j} + \sum_{t} s_{a,t} x_{t,j} = y_{a,j} + 2c_{a,j+1}, \qquad y_{a,j} \in \{0,1\}. \tag{278}$$

Multiply the state weight by the input-bit weights and by $\prod_{a:j<n_a} b_{a,j}(y_{a,j})$, then merge equal carry tuples. After an axis ends, its remaining bits are zero. After an address ends, its remaining output bits and final carry must be zero. These conditions enforce the declared domains rather than wrapping addresses.

For one address, let $S_a = \sum_t s_{a,t}$. At each layer its possible carries lie in an interval of at most $S_a + 1$ integers, by the same calculation as in Lemma A.3. The joint state has at most $\prod_a (S_a + 1)$ carry tuples before any boundary conditions are added; often only a small subset is reachable. If a_j axes have an active bit at layer j, there are at most 2^{a_j} bit choices per state. Theorem A.2 charges those transitions and their local weights. This accounts for the dependence on both the strides and the number of simultaneously active axes.

An axis of arbitrary length can be split into aligned power-of-two intervals, translating the base for each interval. Alternatively, its length condition can be retained as a comparison state. An arbitrary dense axis that has no useful bit factorization can be enumerated, using each coordinate to initialize an affine contraction of the remaining axes. Enumerating several such axes costs the product of their lengths. We use this option only when those lengths are already small enough for the required verifier bound.

A.5 Sharing Equality Work and Combining Families

The preceding bounds separate the cost of the contraction from preparation of its local weights. We now give elementary ways to reduce that preparation and to reuse a contraction across several contributions.

Interval offset equality. For a stored vector $f \in \mathbb{F}^{\ell}$, a scale $s \in \mathbb{F}$, and offset o, consider

$$s \sum_{z=0}^{\ell-1} f[z] \widetilde{\mathsf{eq}}(\mathbf{r}, o + z). \tag{279}$$

Out-of-domain addresses contribute zero. If the clamped interval is empty, the result is zero. Otherwise let $[\mathsf{lo}, \mathsf{hi}]$ be its inclusive endpoints. The little-endian multilinear fold need merge only parents with at least one child whose subtree intersects the specified interval. Its exact number of parent merges is

$$\sum_{j=0}^{n-1} \left(\left\lfloor \frac{\mathsf{hi}}{2^{j+1}} \right\rfloor - \left\lfloor \frac{\mathsf{lo}}{2^{j+1}} \right\rfloor + 1 \right).$$

With two children v_0, v_1, use $v_0 + r_j(v_1 - v_0)$; with one child, treat the other as zero. Each merge needs one multiplication, followed by one final multiplication by s. This gives an evaluator for arbitrary stored interval weights without extending them to the whole vector domain.

Prepared equality factors. When many addresses share the same point, split its coordinates into low and high blocks. Preparing their equality weights takes $O(2^{n_{\mathsf{lo}}} + 2^{n_{\mathsf{hi}}})$ field operations and storage, after which each full equality weight is a product of two lookups. A balanced split reduces the stored weights to square-root size in the full domain. If even that storage is undesirable, prepare only a short block and evaluate the remaining bits on demand. The latter choice retains a linear dependence on the number of unprepared bits per lookup. The same tradeoff applies to products of arbitrary bit weights.

Power-of-two carry peeling. For $x = q + 2^m h$, an offset contributes at most one carry from the low block. An arbitrary vector of low-factor values can be summarized into two buckets in $O(2^m)$ operations once its equality weights are available. If the low weight is itself an equality polynomial, read its bits instead: for each input bit, multiply its logical equality weight by the equality weight of the output address bit. Retaining the addition carry gives a two-state recurrence taking $O(m)$ field operations. This evaluates the paired low weights without enumerating either full vector of weights.

Geometric digit weights. Gadget powers allow another useful saving. Let the digit weights be $g(u) = g(0)\gamma^u$, with $\gamma \neq 0$, and let $(E[v])_{v<Q}$ be a prepared vector of low-block equality weights. Define the weighted prefix sums

$$P(v) := \sum_{z<v} \gamma^z E[z].$$

A digit window beginning at a and ending before Q then satisfies

$$\sum_{u<d} \gamma^u E[a+u] = \gamma^{-a}\big(P(a+d) - P(a)\big).$$

Precompute the needed powers and inverse powers of γ alongside P. Each such window is now evaluated in constant work; multiply by $g(0)$ to recover the original digit weights. If $d \leq Q$ and a window crosses the low-block boundary, split it into two pieces and assign the pieces to their respective carry buckets. Under this condition and for unit digit stride, this reduces the low-summary work from $O(dQ)$ to $O(d + Q)$ for a fixed base residue, including the preparation of these prefix sums. A zero ratio or a non-geometric sequence uses the direct summary instead. No equality-point coordinate is inverted.

Reuse across offsets and additive families. Consecutive-digit families with the same strides, factors, and low base residue have identical low summaries for the same interval of low coordinates. Compute those summaries once. Their different high offsets and scalar weights can then initialize one shared high-state vector. At every layer, summing weights of equal states before continuing is valid by linearity of the recurrence. Likewise, in paired contractions, families with identical remaining transitions can share their carry-state evolution. The cost is the preparation of the initial contributions plus the transitions actually processed; forming and grouping those contributions must also be counted.

Aligned families admit a still simpler combine. If distinct axes occupy disjoint bit ranges in every address and the offsets do not generate carries into those ranges, factor them independently as in Appendix A.1. If the address maps also preserve the separate bits within an axis, a unit or bit-product factor on that axis costs work proportional to its bit length. A dense axis requires reading its entries and evaluating their local weights. When alignment fails, retain the carry recurrence. Each method evaluates the same structured sum $T_A(\mathbf{r}; u) = \widetilde{U}_A(\mathbf{r})$.

B Evaluating Akita's Public Weights

Akita's sum-checks leave public weight polynomials for the verifier to evaluate. The protocol specifies these weights; this appendix gives the calculations that avoid enumerating all their Boolean evaluations. We use the general structured evaluation algorithms of Appendix A whenever the weights factor over logical coordinates but occupy shifted or interleaved vector addresses.

Quotient-free checking requires one additional calculation. Its weights include reduction modulo the native ring polynomial, so ordinary powers of the ring-evaluation point no longer suffice. We first derive the residue kernels and their final sum-check evaluations. We then give the factors for the remaining checks and explain how overlapping setup views are combined, both for a direct scan and after setup offloading.

B.1 Quotient-Free Relation Weights

Fix a native ring dimension d, a public multiplier $A(X) = \sum_{k<d} a_k X^k$, and an evaluation point $\alpha \in E$. The residue kernel of (150) is equivalent to the rotation-matrix reduction used concurrently in the current Grand Danois construction [56]. To make the equivalence exact, define the signed rotation

$$\mathrm{rot}(v_0, \ldots, v_{d-1}) := (-v_{d-1}, v_0, \ldots, v_{d-2}), \tag{280}$$

and let $\mathsf{cf}_d(A) := (a_0, \ldots, a_{d-1})^\top$. The matrix

$$\mathsf{M}_A := \left[\mathsf{cf}_d(A) \mid \mathrm{rot}(\mathsf{cf}_d(A)) \mid \cdots \mid \mathrm{rot}^{d-1}(\mathsf{cf}_d(A))\right] \tag{281}$$

has as its jth column the coefficient vector of the corresponding reduced basis product. Consequently, for $\boldsymbol{p}_d(\alpha)^\top := (1, \alpha, \ldots, \alpha^{d-1})$,

$$\left(\boldsymbol{p}_d(\alpha)^\top \mathsf{M}_A\right)_j = \kappa^{(d)}_{A,\alpha}(j). \tag{282}$$

Thus Grand Danois's projected rotation row and the residue kernel are the same linear map for one ring product. The residue-kernel formulation lets us place that map inside the common row normal form, where different row families may have different native dimensions and noncontiguous witness address maps.

Computing every weight independently from the expanded formula would cost $O(d^2)$ operations. Consecutive basis monomials differ by multiplication by X, so their weights are closely related. We use this relation to compute all d weights in linear time.

Lemma B.1 (Kernel recurrence). *The residue kernel of $A(X) = \sum_{k<d} a_k X^k$ satisfies*

$$\kappa^{(d)}_{A,\alpha}(0) = A(\alpha), \tag{283}$$

$$\kappa^{(d)}_{A,\alpha}(j+1) = \alpha \kappa^{(d)}_{A,\alpha}(j) - (\alpha^d + 1) a_{d-1-j}, \qquad j \in [d-1]. \tag{284}$$

Consequently, one complete kernel is computed in $O(d)$ extension-field operations without dividing by $\alpha^d + 1$.

Proof. The initial value follows because A already has degree less than d. Multiplying $AX^j \bmod (X^d + 1)$ by X shifts every coefficient by one place. The coefficient a_{d-1-j} is the unique new term that crosses degree d. Ordinary evaluation multiplies the shifted polynomial by α, while replacing α^d by -1 subtracts $(\alpha^d + 1)a_{d-1-j}$. This gives (284).

The quotient correction is now public. For each public basis product, Euclidean division gives a public *basis quotient* $q^{\mathsf{basis}}_{A,j}$ such that

$$A(X)X^j = \left(A(X)X^j \bmod (X^d + 1)\right) + (X^d + 1) q^{\mathsf{basis}}_{A,j}(X). \tag{285}$$

Evaluating this identity shows how the quotient contribution enters the kernel:

$$\kappa^{(d)}_{A,\alpha}(j) = A(\alpha)\alpha^j - (\alpha^d + 1) q^{\mathsf{basis}}_{A,j}(\alpha). \tag{286}$$

For the complete row, the quotient of (144) is

$$Q_r(X) = \sum_{c \in \mathsf{Occ}_r} \sum_{j=0}^{d_r-1} w_{\mathsf{addr}_{r,c}(j)} q^{\mathsf{basis}}_{A_{r,c},j}(X). \tag{287}$$

This explains where the private quotient has gone. Its contribution is a linear combination of the witness coefficients with public basis-quotient weights. Incorporating those weights into the check accounts for the same reduction without asking the prover to supply Q_r separately.

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. We combine these with the scalar opening rows using a row-batching point. To write the resulting coefficient of a witness entry, we add every contribution from a row that reads that entry. Let $\mathsf{Rows}_{\mathsf{field}}$ be the ordered set of scalar field rows. Write a row in this set as

$$z_r^{\mathsf{fld}} := \sum_{u \in [N_{\mathsf{w}}]} b_{r,u} w_u - y_r, \tag{288}$$

where the public coefficients $b_{r,u} \in E$ include the direct or trace opening weights. Let n_{rel} be the total number of ring and field rows, let $s_{\mathsf{rel}} := \lceil \log_2 n_{\mathsf{rel}} \rceil$, and pad the row vector with zero rows to length $2^{s_{\mathsf{rel}}}$. The existing row-batching point $\tau_1 \in E^{s_{\mathsf{rel}}}$ supplies

$$\omega_{\mathsf{row},r}(\tau_1) := \widetilde{\mathsf{eq}}(\tau_1, r). \tag{289}$$

For the quotient-free route, define the public coefficient vector on the padded witness domain by

$$\begin{aligned} m^{\mathsf{qf}}_{\alpha,\tau_1}(u) := & \sum_{r \in \mathsf{Rows}_{\mathsf{ring}}} \omega_{\mathsf{row},r}(\tau_1) \sum_{\substack{c \in \mathsf{Occ}_r,\ j \in [d_r] \\ \mathsf{addr}_{r,c}(j) = u}} \kappa^{(d_r)}_{A_{r,c},\alpha}(j) \\ & + \sum_{r \in \mathsf{Rows}_{\mathsf{field}}} \omega_{\mathsf{row},r}(\tau_1) b_{r,u}, \qquad u \in [2^{\mu'}], \end{aligned} \tag{290}$$

where every summand with $u \geq N_{\mathsf{w}}$ is zero. Its public target is

$$V^{\mathsf{qf}}_{\alpha,\tau_1} := \sum_{r \in \mathsf{Rows}_{\mathsf{ring}}} \omega_{\mathsf{row},r}(\tau_1) Y_r(\alpha) + \sum_{r \in \mathsf{Rows}_{\mathsf{field}}} \omega_{\mathsf{row},r}(\tau_1) y_r. \tag{291}$$

These definitions give (154). Equation (290) sums all uses of a witness coordinate. It therefore remains valid when two row families read overlapping witness coordinates under different native dimensions.

Verifier evaluation at the sum-check point. The prover may enumerate the coefficient vector in (290) directly. A succinct verifier instead needs its multilinear extension only at the final sum-check point $\mathbf{r}_x$. For one occurrence (r, c), define its exact equality weights by

$$e_{r,c,\mathbf{r}_x}(j) := \widetilde{\mathsf{eq}}\big(\mathbf{r}_x, \mathsf{addr}_{r,c}(j)\big), \qquad j \in [d_r]. \tag{292}$$

Here, as elsewhere in the paper, $\widetilde{\mathsf{eq}}(\mathbf{r}_x, u)$ with an integer second argument means the equality polynomial evaluated at the μ'-bit encoding of address u. This definition uses the witness address map and does not assume that the native window starts at zero or occupies a global tensor coordinate.

Transposing the sum in (151) once more moves these equality weights onto the public coefficients of A. For any sequence $e = (e_0, \ldots, e_{d-1}) \in E^d$, define the *transposed residue kernel*

$$\kappa^{(d),\top}_{e,\alpha}(k) := \sum_{j=0}^{d-1} e_j (-1)^{\lfloor (k+j)/d \rfloor} \alpha^{(k+j) \bmod d}. \tag{293}$$

The desired MLE contribution is then

$$\sum_{j=0}^{d-1} e_j \kappa_{A,\alpha}^{(d)}(j) = \sum_{k=0}^{d-1} a_k \kappa_{e,\alpha}^{(d),\top}(k). \tag{294}$$

The terminal kernel obeys the same wraparound recurrence:

$$\kappa_{e,\alpha}^{(d),\top}(0) = \sum_{j=0}^{d-1} e_j \alpha^j, \tag{295}$$

$$\kappa_{e,\alpha}^{(d),\top}(k+1) = \alpha \kappa_{e,\alpha}^{(d),\top}(k) - (\alpha^d + 1) e_{d-1-k}, \qquad k \in [d-1]. \tag{296}$$

The proof is Lemma B.1 with the two coefficient sums exchanged. The verifier prepares one such kernel for each distinct native address geometry and takes a dot product against every public multiplier with that geometry. This is the quotient-free analogue of the power tensor $\widetilde{\alpha}$ in (49).

B.2 Opening, Range, and Norm Weights

We now apply the structured evaluator to the opening, range, and norm checks. Each application specifies the logical weights, the witness address, and the domain included in the sum. Once those are fixed, the algorithms of Appendix A determine the verifier's work without expanding the corresponding witness-sized vector of weights.

Opening rows. For direct coefficient packing, the public factors are those of (127). We place them at the actual addresses of the opening digits before evaluating their multilinear extension. Let o_e be the first flat coefficient of the opening-digit interval and write $s_e = kd_{\mathsf{car}}/d_D$. 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 gives

$$\mathsf{addr}_e(i, \ell, t, j) := o_e + \big((i s_e + v)\delta_{\mathsf{open}} + \ell\big) d_D + u. \tag{297}$$

The relation sum-check uses the multilinear extension of this row. At its final point $\mathbf{r}_x$, the verifier needs only

$$\widetilde{\omega}_{\mathsf{dir}}(\mathbf{r}_x) = \sum_{i,\ell,t,j} \omega_{\mathsf{dir}}(i, \ell, t, j)\, \widetilde{\mathsf{eq}}\big(\mathbf{r}_x, \mathsf{addr}_e(i, \ell, t, j)\big). \tag{298}$$

All weights in this sum are public. When the opening-digit interval is bit-aligned with power-of-two digit depth, the equality weight factors along the block, native-subcolumn, digit, and native-coefficient axes. Otherwise the offset-equality evaluation of Proposition A.4 accounts for the carries in the address formula. This lets the verifier evaluate the row at $\mathbf{r}_x$ without constructing its $B\delta_{\mathsf{open}} k d_{\mathsf{car}}$ entries.

Evaluation-trace weights. Let $\tau_\nu := \mathcal{T}_{\rho_{\mathsf{pack}}}(X^\nu)$ for $0 \le \nu < d_A$, with the functional of Section 7.1. Recomposition gives the logical row

$$\omega_{\mathsf{Tr}}(i, \ell, \nu) = \theta \widetilde{\mathsf{eq}}(\rho_{\mathsf{blk}}, i) b_{\mathsf{op}}^{\ell} \tau_\nu.$$

Here θ is the checked tensor-reduction scale; take $\theta = 1$ for an unscaled claim. This row is placed at the opening-digit addresses and is zero elsewhere. Its coefficient factors τ_ν depend on the opening point and packing, but not on the block index or witness.

First suppose the opening interval is bit-aligned, B is a power of two, and the digit axis of each native block is padded from depth δ_{open} to $\Delta = 2^{\lceil \log_2 \delta_{\mathsf{open}} \rceil}$ with zero weights. Its witness address bits consist of a fixed interval tag, block index, native subcolumn, digit index, and native coefficient. Split the query point r accordingly and collect the subcolumn and native-coefficient coordinates into r_{coef}, in the order of ν. Separating the factors gives

$$\begin{aligned} &\widetilde{\omega}_{\mathsf{Tr}}(r) = \theta \widetilde{\mathsf{eq}}(r_{\mathsf{tag}}, \mathsf{tag}) \widetilde{\mathsf{eq}}(\rho_{\mathsf{blk}}, r_{\mathsf{blk}}) D(r_{\mathsf{digit}}) C(r_{\mathsf{coef}}), \\ &D(s) := \sum_{\ell < \delta_{\mathsf{open}}} b_{\mathsf{op}}^{\ell} \widetilde{\mathsf{eq}}(s, \ell), \qquad\qquad C(t) := \sum_{\nu < d_A} \tau_\nu \widetilde{\mathsf{eq}}(t, \nu). \end{aligned} \tag{299}$$

In particular, the block factor follows from

$$\sum_{i\in\{0,1\}^{\log B}} \widetilde{\mathsf{eq}}(\rho_{\mathsf{blk}}, i)\widetilde{\mathsf{eq}}(r_{\mathsf{blk}}, i) = \prod_j \big((1-\rho_{\mathsf{blk},j})(1-r_{\mathsf{blk},j}) + \rho_{\mathsf{blk},j} r_{\mathsf{blk},j}\big)$$
$$= \widetilde{\mathsf{eq}}(\rho_{\mathsf{blk}}, r_{\mathsf{blk}}).$$

Thus no iteration over blocks is needed. After computing τ within one ring, the verifier evaluates the tag and block factors in $O(\log|\mathbf{w}'|)$ field operations. Generating the Boolean equality tables recursively evaluates D and C in $O(\delta_{\mathsf{open}} + d_A)$ operations, including zero padding to $\Delta < 2\delta_{\mathsf{open}}$. This proves the claimed $O(\log|\mathbf{w}'| + \delta_{\mathsf{open}} + d_A)$ evaluation cost for this layout; the ring-local preprocessing is independent of B.

Actual witness intervals may be shifted, block ranges truncated, and digit axes stored without power-of-two padding. The simple product above then need not hold. Use the actual witness address map instead:

$$\widetilde{\omega}_{\mathsf{Tr}}(r) = \theta \sum_{i,\ell,\nu} \widetilde{\mathsf{eq}}(\rho_{\mathsf{blk}}, i) b_{\mathsf{op}}^{\ell} \tau_{\nu} \widetilde{\mathsf{eq}}(r, \mathsf{addr}_e(i,\ell,\nu)).$$

The logical block weight and equality weight at the witness address give the paired form (273). Contract the coefficient traces within their native ring dimension, then apply the affine or paired carry calculation of Appendix A to the remaining block, lane, and digit coordinates. Separate witness intervals contribute additively, with their own offsets and block ranges.

Restricted equality and response norm weights. For the binary support $\mathcal{I}_{\mathsf{bin}}$ of Section 6.1, the required weight is

$$\sum_{i\in\mathcal{I}_{\mathsf{bin}}} \widetilde{\mathsf{eq}}(\mathbf{s}, i)\widetilde{\mathsf{eq}}(\mathbf{r}, i).$$

Represent each disjoint interval $[o, o+L)$ included in the sum by one logical coordinate $x \in [L]$ and the two maps $A_L(x) = A_R(x) = o + x$. Its contribution is a paired affine contraction with unit logical weights. Sum the interval contributions. This evaluates the multilinear extension of the pointwise product on Boolean addresses; multiplying the separate multilinear extensions of the support indicator and the equality weights would give a different polynomial.

The norm-linking identities in Section 6.2 have the same form with different maps. One address identifies a coefficient of a reconstructed response or one of its digit planes; the other identifies the corresponding committed witness cell. Recomposition powers supply the digit factors. The paired recurrence contracts these maps, while the intervals determine which response cells participate. Thus the cost is controlled by the factor descriptions, intervals, and carry states, rather than by the size of the padded witness domain.

B.3 Factored Blocks in the Relation

Each factored block in Figure 6 uses its native address map from Section 5.5, including its segment offset. For example, a block with dimension d, width s, digit depth δ, and offset o has address $A(j,y,u,k) = o + d\delta sj + d\delta y + du + k$. The coefficient lane k changes first; consecutive digit planes have stride d. We contract these factors using the separate-stride and several-address-map extensions of Appendix A.4, retaining all four coordinates. The scalar consecutive-digit map (276) is only the $d = 1$ specialization.

Coefficient packing uses the following factors.

– On $\hat{\mathbf{e}}$, the carrier-consistency weight for cell (i,ℓ,t,j) is

$$c_i(\alpha)\,\beta_t\,b_{\mathsf{op}}^{\ell}\,\alpha^j.$$

The challenge evaluation is shared across all k coordinate planes.

– On $\hat{\mathbf{z}}$, the carrier row uses the direct contraction: the public factors are the position weight $\widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{pos}}, x)$, the low-coefficient weight $\widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{pack}}, a)$, the carrier power α^j, and the two gadget digits, with ambient coefficient address $a + khj$.

– On $\hat{\mathbf{t}}$, ambient row r uses $c_i(\alpha^{kh})$ together with the $\mathbf{A}$-row selector and the opening gadget digit. This block 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 (298).
– On the quotient-lift route, a carrier quotient cell in coordinate plane t and carrier position j has basis-and-power factor $\beta_t \alpha^j$, scaled by $-(\alpha^{d_{\mathsf{car}}} + 1)$ and its quotient gadget digit. Every other quotient family uses its own native power and $-(\alpha^d + 1)$.
– On the quotient-free route, these quotient factors are absent. The corresponding cells instead use the public transpose-convolution weights of Section 7.3. This formula completes the common evaluator; the admitted schedule family invokes it only after the switch to evaluation trace, never at a coefficient-packing fold (Definition 9.2).

Evaluation trace replaces the coefficient-packing factors 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 ordinary 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 (132). For group g, multiply the unscaled scalar-row weights $\omega_{\mathsf{Tr},g}$ by the public scalar $\vartheta_{g,c}\theta_g$ from (134). Here θ_g includes the group's tensor accumulator at its native prefix and the equality factor on any extra variables in the shared reduction; it is one for the identity reduction. On the quotient-lift route, the evaluation-trace fold quotient uses the ambient modulus factor $-(\alpha^{d_A} + 1)$.

On the quotient-free route, the same public multipliers act through the residue kernels of (150), not through their ordinary evaluations followed by a common α-power. At the final sum-check point, the verifier contracts each native coefficient occurrence with its exact equality sequence from (292) and uses (294). There is no additional common $\widetilde{\alpha}$ factor: the terminal residue kernel already contains the complete coefficient-coordinate contraction.

The compression rows use the same procedure for their gadget and ring-reduction terms, restricted to the scheduled digit intervals. Their matrix terms use directly evaluated $\mathbf{F}/\mathbf{H}$ matrices, with matrix evaluations shared across identical shapes. These smaller maps remain local to the verifier, including at an offloaded level. They do not enter the combined $\mathbf{A}/\mathbf{B}/\mathbf{D}$ scan described next.

B.4 Shared Setup Views and Offloaded Weights

The setup blocks are different in kind: a matrix coefficient is sampled from the shared setup rather than assembled from small public factors. Let $\mathcal{M}_{\mathsf{setup}}$ index the active instances of $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{D}$. The $\mathbf{F}/\mathbf{H}$ compression matrices are evaluated separately. An instance $I \in \mathcal{M}_{\mathsf{setup}}$ has shape (n_I, m_I, d_I) and reads the flat prefix of length $n_I m_I d_I$. Write $N_{\mathsf{active}} := \max_{I \in \mathcal{M}_{\mathsf{setup}}} n_I m_I d_I$ for the longest of these prefixes. A single setup coefficient can be used by several matrix instances, so we first identify its position in each view. For a flat index p in instance I's prefix, define
$$\mathsf{row}_I(p) := \left\lfloor \frac{p}{m_I d_I} \right\rfloor, \quad \mathsf{col}_I(p) := \left\lfloor \frac{p \bmod (m_I d_I)}{d_I} \right\rfloor, \quad \mathsf{coef}_I(p) := p \bmod d_I.$$
Thus the same flat coefficient can acquire different monomial or residue weights in views with different ring dimensions. No global setup ring dimension is used.

To combine the views, we add their weights at each flat index p. Multiplying the resulting weight $\bar{\omega}(p)$ by S_p accounts for all uses of that coefficient. Both reduction routes give the inner product (159). The coefficient weight $\bar{\omega}(p)$ depends on the selected reduction. For quotient lifting, it is
$$\bar{\omega}^{\mathsf{ql}}(p) := \sum_{\substack{I \in \mathcal{M}_{\mathsf{setup}} \\ p < n_I m_I d_I}} \Omega_I(p)\, \alpha^{\mathsf{coef}_I(p)}, \tag{300}$$
where $\Omega_I(p)$ contains the row-batching weight at $\mathsf{row}_I(p)$ and the structured witness-column weight at $\mathsf{col}_I(p)$.

For quotient-free checking, a coefficient can contribute through several witness windows even within one matrix view. In particular, the same $\mathbf{B}$ matrix is used for several logical slices, each with its own witness interval and public weight. The $\mathbf{A}$ and $\mathbf{D}$ maps are unsliced. We must include each use with the equality weights of its own window.

Let $\mathsf{Occ}_I(p)$ index the logical row-and-witness-window occurrences in which coefficient p of view I participates. For an occurrence $\nu \in \mathsf{Occ}_I(p)$, let $e_{I,\nu,\mathbf{r}_x}$ be the exact equality sequence of (292), and let $\Omega_{I,\nu}(p)$ contain all public row, gadget, slice, and projection factors other than the native coefficient functional. The quotient-free weight is

$$\bar{\omega}^{\mathsf{qf}}(p) := \sum_{\substack{I \in \mathcal{M}_{\mathsf{setup}} \\ p < n_I m_I d_I}} \sum_{\nu \in \mathsf{Occ}_I(p)} \Omega_{I,\nu}(p) \kappa^{(d_I),\top}_{e_{I,\nu,\mathbf{r}_x},\alpha}\big(\mathsf{coef}_I(p)\big). \tag{301}$$

This is (294) applied to every matrix in $\mathcal{M}_{\mathsf{setup}}$. For $\mathbf{B}$, the occurrence sum includes the logical slices whose interval before padding contains the column; padding contributes zero. The combined weight accounts for overlapping setup views and lets the verifier scan only their longest prefix.

For shared slices, the underlying linearity is already visible in the ring rows. If $\beta_{s,r}$ weights logical row (s, r), their combined $\mathbf{B}$ contribution is

$$\sum_{r \in [n_B]} \sum_{c \in [m_B]} B_{r,c} \left(\sum_{s \in [S_B]} \beta_{s,r} x_{B,s,c} \right). \tag{302}$$

At the final sum-check point, the corresponding public weights are combined in the same way, with the coefficient functional selected by the reduction route as above.

There is still work to do before multiplying by each setup coefficient: the verifier must compute its combined weight. For $\mathbf{B}$, explicitly listing every combination of a slice and a column could make this work expensive. We therefore contract the slice weights using their structured addresses.

For an unsliced matrix, Proposition A.4 computes column weights in amortized constant field work per column for fixed digit counts, after preparing the local equality weights. For a sliced matrix, the slice number, its exact interval, and the consecutive-digit address form one additional structured factor. The evaluator of Appendix A contracts these factors with the same carry state. A direct fallback costs $O(S_B)$ field operations per matrix entry; the structured route has the exact transition cost of Theorem A.2.

The Setup Weight after Offloading At the end of setup offloading, the verifier must evaluate $\widetilde{\omega}_{\mathsf{setup}}(\rho_S)$ (Section 9.2). Expand this public weight by its logical matrix contributions. A typical family has the form

$$\sum_{x} \left(\prod_{t} u_t(x_t) \right) \widetilde{\mathsf{eq}}(\rho_S, A_{\mathsf{setup}}(x)) \widetilde{\mathsf{eq}}(\mathbf{r}_x, A_{\mathsf{witness}}(x)).$$

Here x ranges over the row, column, digit, and projection-lane coordinates included in the sum. The first map locates a flat setup coefficient; the second locates the witness coordinate whose public weight multiplies it. The factors include row-batching weights and the ring-reduction powers. This is a paired affine contraction. Native coefficient bits that align with every contribution can be factored off first; differing ring dimensions leave additional lane coordinates with geometric weights.

For sliced $\mathbf{B}$, intersect each logical slice with the relevant interval of source blocks and use the resulting offset and length. Several slices can refer to the same matrix coefficient, so their contributions add. The $\mathbf{A}$ and $\mathbf{D}$ families are unsliced. Summing these families evaluates the public setup weight without reading the setup coefficients. The setup value itself is authenticated by the carried opening; no evaluator here assumes that the pseudorandom setup entries have a small tensor description.

The constant-root analysis in Section 9.4 uses these bounds with controlled factor descriptions and address states. In particular, long equality factors should be supplied through their bit products, whereas folding challenges supplied explicitly are charged by their vector length. For concrete parameters, the verifier can choose between an aligned factorization, a carry contraction, and direct evaluation of a small factor, accounting in each case for both preparation and contraction.

C SIS Security Estimation

Akita sizes every matrix through the Module-SIS problem of Definition 3.4. 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 using the ADPS16 model. For a lattice-reduction block size β, this model assigns the quantum attack cost $2^{0.2650\beta}$. The estimator searches over the attack parameters and reports the cheapest attack it finds. For the concrete parameter family studied here, we require every Module-SIS instance to have estimated quantum ADPS16 attack cost at least 2^{128}. This common per-instance floor applies to outer commitments $\mathbf{B}_j$, opening commitments $\mathbf{D}_{\mathsf{batch}}$, inner commitments $\mathbf{A}$, and every compression map $\mathbf{F}_i, \mathbf{H}_i$.

The security proof must still compose these instance-wise assumptions. Let c_I denote 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

$$\lambda_{\mathsf{fam}} := -\log_2\left(\sum_{I \in \mathbb{I}_{\mathcal{P}}} 2^{-c_I}\right), \tag{303}$$

where $\mathbb{I}_{\mathcal{P}}$ contains each distinct shared-prefix matrix instance induced by that family, as in Corollary 10.20. Under the quantum ADPS16 Core-SVP model, the direct audit reported in the artifact gives $\lambda_{\mathsf{fam}} > 125$ bits of quantum security; every fixed schedule is stronger, with the weakest audited schedule exceeding 127 quantum bits [65]. We state the lower 125-bit quantum figure because it also covers an adversary that chooses a schedule from that entire public family after seeing the shared setup.

We do not regard the gap between 128 quantum bits per instance and 125 bits of family-wide quantum security as a security concern. The quantum ADPS16 Core-SVP model is already conservative: it gives the attacker the idealized quantum shortest-vector cost and omits the surrounding cost of carrying out BKZ. The union bound is conservative in a different direction. It adds the modeled advantages of many heterogeneous matrix instances, but it neither constructs nor corresponds to a single attack that amortizes work across those instances. Thus the family-wide loss is proof-level accounting, not evidence of a 2^{125}-work quantum attack on Akita.

The inventory is not a claim that every level has every matrix. In particular, the Akita terminal uses a predecessor-bound inner $\mathbf{t}$ state and induces its terminal $\mathbf{A}$ collision instance; it has no $\mathbf{B}$, $\mathbf{D}$, $\mathbf{F}$, or $\mathbf{H}$ row. For comparison, ADPS16 assigns the classical attack cost $2^{0.2920\beta}$; at a 128-bit quantum floor this is about 2^{141}, the same per-instance floor expressed in classical terms rather than a separate acceptance rule. Coefficient-ℓ_∞ instances use the direct infinity-norm attack equations with LGSA for the reduced-basis shape. Here LGSA is the L-shaped geometric-series-assumption profile of `simulator.LGSA` in `lattice-estimator`: 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 [65]. Euclidean instances use the Euclidean attack equations, but retain the same quantum ADPS16 cost model and the same 128-bit per-instance floor for this parameter family. The norm choice is part of the instance supplied by the reduction, not an estimator-side conversion.

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. In practice, the attacker first runs BKZ to improve the basis. Its main parameter is the block size β: a larger block lets BKZ find shorter vectors, but requires solving a harder shortest-vector problem inside each block. An estimator must therefore answer two different questions. It first predicts how large β must be before the desired collision becomes visible in the reduced basis. It then predicts how much work an attacker pays for reduction at that block size.

The first prediction depends on the collision norm and on the simulated shape of the reduced basis. The second is the *reduction-cost model*. ADPS16, BDGL16, and MATZOV are three answers to this second question. None changes the SIS problem or the short vector that breaks binding; they change what work is counted after the estimator has determined β. Thus, "128 bits" does not mean that a theorem forces every attack to perform exactly 2^{128} machine instructions. It means that the cheapest attack found by the estimator costs at least 2^{128} work units under the stated basis-shape and reduction-cost models.

At a high level, ADPS16 prices only the hardest shortest-vector call inside BKZ; BDGL16 prices 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

$$\text{quantum ADPS16} < \text{classical ADPS16}$$
$$< \text{classical MATZOV} < \text{classical BDGL16}.$$

The first inequality reflects the assumed quantum speedup. After that, moving to the right charges the classical attacker for more of the computation around the shortest-vector call. Neither change makes the underlying SIS instance harder.

ADPS16: the Core-SVP convention. ADPS16 uses the simplest accounting. It treats the dominant work in BKZ as one shortest-vector computation in dimension β, and maps β directly to an attack cost. The classical convention assigns $2^{0.2920\beta}$ work and the quantum convention assigns $2^{0.2650\beta}$ work [5]. These formulas intentionally suppress the cost of preparing the basis, invoking the shortest-vector routine many times during BKZ, and managing the required memory. The output is therefore a Core-SVP yardstick, not a gate count, running time, or estimate of the number of physical qubits.

We use the quantum ADPS16 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 reduces 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 this as the hard production gate because its transfer to an end-to-end BKZ attack requires strong assumptions about reusable quantum memory and repeated sieving. We did nonetheless test it as a sensitivity model: an independently optimized sweep found no accepted ADPS16 row below the corresponding rescaled review line.[13]

BDGL16: classical BKZ accounting. BDGL16 begins from a classical lattice sieve, an exponential-time algorithm that stores many lattice vectors and searches for combinations that produce shorter ones. Its asymptotic shortest-vector cost is again approximately $2^{0.292\beta}$ [15]. The implementation in `lattice-estimator`, however, does not report only this exponent. It adds a concrete offset for the sieve, the cost of an initial LLL basis-reduction pass, and an estimate of how many shortest-vector calls BKZ makes. In the common regime $\beta < d$, where d is the lattice dimension, the last factor is modeled as $8d$ calls. This makes BDGL16 a more detailed classical estimate, but 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 [77]. When `lattice-estimator` uses "MATZOV" as a reduction-cost model for SIS, it uses the latter refinement; it is not running the report's full LWE attack. This model prices a progressive BKZ computation, in which the attacker pays for the 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 both β and the ambient lattice dimension, as well as on the selected classical nearest-neighbor backend. In particular, MATZOV credits the attacker with "dimensions for free": it prices 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. There is no single MATZOV exponent that can be substituted for 0.2650 or 0.2920.

Comparison of reduction-cost models. The ordering above is not a theorem for every β and d. ADPS16 depends only on β, whereas MATZOV and BDGL16 also depend on the ambient lattice dimension, and their estimates can cross at small block sizes. At the common reference point $\beta = 500$ and $d = 1024$, however, the current `lattice-estimator` formulas report 132.5 bits for quantum ADPS16, 146.0 for classical ADPS16, 169.7 for classical MATZOV, and 175.4 for classical BDGL16 [4,5,15,77]. MATZOV obtains its middle value by pricing a progressive BKZ run while reducing the effective sieve dimension 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 disagree one step earlier. A basis-shape model such as GSA or LGSA can predict a different required block size β for the same SIS instance; ADPS16, BDGL16, or MATZOV then assigns

[13] The model comparison, assumptions, and sweep are recorded in Akita's SIS policy specification [65].

a cost to that β. RoKoKo's released estimator uses GSA followed by classical MATZOV accounting. The Orthus comparison reports both classical ADPS16 and BDGL16, while Akita's production gate uses LGSA followed by quantum ADPS16 [4,77,5,15]. These are different lenses, so their reported bit counts should not be read as if they measured 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 prices 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 native security lens and any normalization separately in Section 13. A different estimator is not by itself an implementation error: Appendix F 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 $p \in \{2, \infty\}$. 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, developed following the methodology and Python/Sage implementation of the public `lattice-estimator`, with matching formulas where the two models agree [4,65]. The generator then searches for the cheapest attack under the fixed norm, shape, and reduction-cost model and records the smallest rank that reaches the required floor.

The Rust implementation achieves a speedup of approximately three orders of magnitude over the Python/Sage version, making a complete sweep over the thousands of cells in each production profile practical. This speedup does not replace validation of the estimates. 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 include 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 under 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.

Norms and collision buckets. 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 (208). A raw 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 use the ℓ_1 challenge radius, or it may reverse 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 is never applied 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

$$n_{\mathsf{sc}} = rd, \qquad m_{\mathsf{sc}} = m_A d, \qquad L_{\mathsf{sc}} = \sqrt{C_2}, \tag{304}$$

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. The input width therefore appears only in m_{sc}. Multiplying C_2 by m_A or d would count those coordinates twice.

The shipped Euclidean table is keyed by the squared quantity C_2, not by its square root. Given a raw bound $C_2 > 0$, lookup uses the least supported power-of-two bucket $\widehat{C}_2 \geq C_2$. For each modulus profile, ring dimension, module rank, and bucket, a row records the largest supported ring width m_A. The planner chooses the first rank whose recorded width is at least the requested m_A. The upward bucket 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 (304).

Commitment compression (Section 4.5) adds one query per map of each two-stage pair. Each map is priced at its native 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 5. The original $\mathbf{B}, \mathbf{D}$ matrices retain their own MSIS floors, since the extraction of Lemma 4.4 still walks through them.

Concrete compression estimates. For the two-stage parameters of Table 5, we estimate each map at coefficient collision bound one. At the maximum source size of 8 KiB, the first-stage ring widths are 1024, 2048, and 4096 over the respective 32-, 64-, and 128-bit moduli in that table. Scalarization gives 65536 input coordinates in each case, with output dimensions 64, 32, and 16, respectively. The second-stage instances have 2048 input coordinates and output dimensions 32, 16, and 8. 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. The minimizing block sizes are 556 and 578, respectively, yielding these costs through 0.2650β. Smaller supported source images use prefixes of the same first-stage maps; 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 estimates concern the individual compression maps; they enter the combined accounting in (303) alongside the other commitment instances.

Using the estimates in parameter selection. The planner uses these estimates to select candidate ranks. 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 [65,64].

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	Collision norm, 2 or ∞
η	Input	Certified collision norm
n	Output	Least module rank meeting the security floor

D Certifying Operator-Norm Challenge Filters

Akita uses operator-norm rejection sampling for sparse folding challenges in some Euclidean security profiles. This requires two guarantees. First, the integer predicate replayed by the verifier must never accept a challenge whose true multiplication norm exceeds the configured threshold. Second, its accepted set must remain large enough for the security analysis. We prove these properties separately. The first follows from a fixed-point enclosure of every spectral value. The second follows from exact spectral moments and a rational polynomial tail certificate.

Comparison with PikkuFold. PikkuFold [85, 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. Here we supply two further certificates for specified challenge families: an exact arithmetic test that never exceeds the operator-norm threshold, and a proved lower bound on the number of challenges that pass that test. Our mixed-magnitude shell fixes the number of coefficients of each magnitude; it is a fixed-composition subset of PikkuFold's bounded fixed-weight family.

D.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

$$\rho(c) := \|M_c\|_{2\to 2} = \max_{0\leq k<d} |c(\zeta_k)|, \qquad \zeta_k = e^{(2k+1)\pi i/d}. \tag{305}$$

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 by interval 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. Thus the table entries and their error bounds can be verified using exact rational and integer arithmetic.

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

$$\Delta^2|c(\zeta_k)|^2 \leq U_k := R_k^2 + I_k^2 + 2(|R_k| + |I_k|)r + 2r^2. \tag{306}$$

For a positive rational threshold T, the fixed-point predicate accepts exactly when $U_k \leq T^2\Delta^2$ for one frequency from each conjugate pair. Clearing the denominator of T makes this an integer comparison. Write $\mathcal{C}_T^{\mathsf{fp}}$ for the resulting subset of the raw shell $\mathcal{C}_0$.

The same enclosure also identifies a true-norm subset that is certainly accepted. Choose a nonnegative integer h with

$$h^2 > 8r^2, \qquad h < \Delta T, \qquad \tau_T := T - \frac{h}{\Delta}. \tag{307}$$

For a fixed shell, L and hence r are constant; the second condition ensures $\tau_T > 0$.

Theorem D.1 (Fixed-point sandwich). *The fixed-point predicate satisfies*

$$\{c \in \mathcal{C}_0 : \rho(c) < \tau_T\} \subseteq \mathcal{C}_T^{\mathsf{fp}} \subseteq \{c \in \mathcal{C}_0 : \rho(c) \le T\}. \tag{308}$$

Proof. The right inclusion follows directly from (306). 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.

D.2 An Exact Accepted-Support Certificate

Let a, b be nonnegative integers with $1 \le a + b \le d$. Choose uniformly a shell element with exactly a coefficients of magnitude one and b of magnitude two, with independent uniform signs. Writing $w = a + b$, its raw cardinality is

$$N_{\mathsf{raw}} = \binom{d}{w}\binom{w}{b} 2^w. \tag{309}$$

For one frequency, let $X := |c(\zeta_k)|^2$. All $d/2$ representative spectral magnitudes 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 the shell by a signed permutation of coefficient positions. Also $0 \le X \le B := \|c\|_1^2$.

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

$$M_m = \frac{(m!)^2}{N_{\mathsf{supp}}}[s^m t^m u^a v^b] \prod_{j=0}^{d-1} \left(1 + u\cosh(s\alpha^j + t\alpha^{-j}) + v\cosh(2s\alpha^j + 2t\alpha^{-j})\right). \tag{310}$$

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. 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.

A rational tail majorant. Let Q be a rational polynomial of degree at most 30 satisfying

$$Q(x) \ge 0 \quad \text{on } [0, B], \qquad Q(x) \ge 1 \quad \text{on } [\tau_T^2, B]. \tag{311}$$

Then $\mathbf{1}_{\{X \ge \tau_T^2\}} \le Q(X)$ pointwise. If $Q(x) = \sum_i q_i x^i$, its marginal tail is therefore bounded by the exact rational number

$$\Pr[X \ge \tau_T^2] \le q_1^\star := \mathbb{E}[Q(X)] = \sum_i q_i M_i. \tag{312}$$

We search numerically for a polynomial and then round it to rational coefficients. Numerical optimality is not part of the proof: any rational Q satisfying (311) gives a valid certificate.

The two inequalities in (311) are checked 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 has no root in the interval and checks a positive endpoint. A successful check proves the required pointwise inequalities over the full real intervals without floating point. These are sufficient certificate tests; a valid nonnegative polynomial with a zero need not pass the strict-positivity Sturm test just described.

Theorem D.2 (Accepted-support floor). *For a polynomial Q satisfying (311),*

$$\Pr[\rho(c) < \tau_T] \geq p_0 := 1 - \frac{d}{2} q_1^\star. \tag{313}$$

Consequently, $|\mathcal{C}_T^{\mathsf{fp}}| \geq N_{\mathsf{raw}} p_0$. 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 event has probability at most $q_1^\star$ by (312), so a union bound proves (313). The cardinality claim follows from the left inclusion in (308) and uniformity of the raw shell. The trial count is geometric with success probability at least p_0, and all accepted elements have the same probability at each draw.

D.3 Concrete Certificates

We take $\nu = 48$ and the conservative error bound $\varepsilon = 4$ for both concrete dimensions. The trigonometric table construction certifies this error bound; the spectral-moment and polynomial certificates then use it to establish the containment margin and accepted-support floor. The resulting certificates are summarized below. The decimals are rounded displays of exact rational probabilities and their logarithms; the support claims are checked by the rational inequality $N_{\mathsf{raw}} p_0 \geq 2^{128}$.

Table 12: Certified operator-norm challenge filters.

d	shell (a, b)	T	h	$q_1^\star$	p_0	$\log_2(N_{\mathsf{raw}} p_0)$
64	$(31, 11)$	18	600	0.02390904	0.23491065	128.062439
128	$(31, 0)$	13	351	0.00812072	0.48027391	128.563317

For $d = 64$, the shell 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 shell 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 [65]. These certificates bound the cardinality of a fixed, witness-independent challenge family. Pairwise differences remain invertible because the filter only removes elements from the chosen raw 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.

E Akita in Jolt

Jolt's PIOP produces evaluation claims about logical columns: instruction and memory addresses, increment digits, advice, and committed program data. We implement these columns using a small collection of commitment objects. Columns belonging to the execution trace share one prefix-packed polynomial; advice and committed-program objects retain 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 and its catalog-based setup-offloading extension. The interface includes multi-point advice openings and joint proof framing [11]; Appendix I records the exact Jolt and Akita revisions used in the measurements.

E.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

$$W_{\mathsf{trace}}(\mathbf{z}, \mathbf{x}) = \sum_{i=0}^{C-1} \widetilde{\mathsf{eq}}(\mathbf{z}, \mathsf{bin}_s(i))\, W_i(\mathbf{x}). \tag{314}$$

Thus one commitment binds all trace columns, and the polynomial's arity is $s + t + a$.

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. Application semantics, however, refer only to the restrictions to the used slots $W_0, \ldots, W_{m-1}$. The extracted polynomial need not be globally zero on unused slots; the opening reduction below accounts for their contribution directly.

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 handle. It is not 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. These parameters are frozen before its eventual opening point is known, as in Section 4.6. Here, precommitted refers to this order relative to the trace commitment; it does not mean that untrusted advice was known during preprocessing.

E.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

$$W(\boldsymbol{\zeta}, \mathbf{r}) = y_{\mathsf{pack}}, \qquad y_{\mathsf{pack}} := \sum_{i=0}^{m-1} \widetilde{\mathsf{eq}}(\boldsymbol{\zeta}, \mathsf{bin}_s(i))\, y_i. \tag{315}$$

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 E.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 (315) 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 E.1. For several objects, these conditional algebraic errors add over their selector draws; their Fiat–Shamir accounting is separate from the backend PCS error. The application uses the extracted restrictions to the used slots and does not require a proof that the unused region is identically zero. Booleanity, one-hot validity, and the relations connecting decoded values to the execution remain obligations of the Jolt PIOP.

E.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, while all earlier groups retain 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. It 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 raw 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 therefore not the size of the entire Jolt proof.

E.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 owned by the application. It 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. Restoring a transported setup reconstructs and validates its catalog before use. Its digest is bound into the setup transcript, and the proof selects a row from this catalog rather than supplying new schedule parameters.

Guided adaptation retains the scalar trace row's recursive depth, dimensions, block geometry, opening parameters, relation modes, and direct-versus-offloaded topology. It rederives the grouped root's shared opening matrix and every dependent witness length, rank, response bound, setup-prefix length, and proof cost. The adapted row passes the ordinary schedule and catalog admission checks. If the retained structure cannot accommodate the new objects, preprocessing fails; the guide does not claim the optimum 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.

The choice of where offloading becomes worthwhile is an evaluation policy. The catalog-based integration selects it through the precomputed catalogs, while 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.

E.5 Transcript Grinding in the Backend

The Akita dependency used by the merged Jolt integration already implements two distinct nonce mechanisms.[14] 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 gives the classical online random-oracle work bound for this mechanism. It does not provide a QROM theorem or a complete end-to-end security estimate; the remaining Jolt and Akita errors and computational reductions must also be accounted for.

A schedule-derived plan fixes the query order and nonce widths, and its digest is bound into the instance descriptor. Both nonce mechanisms use one packed proof-level stream whose replay must read exactly the entries specified by the plan. 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.

F 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. In particular, none of the implementation findings below is presented as an end-to-end attack on the complete proof system.

F.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 [79, 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 [79, 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

$$T_y := f_y(\mathbf{r}_{\mathsf{tail}}) \in E$$

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:

$$v = \sum_{y \in \{0,1\}^\kappa} \widetilde{\mathsf{eq}}(\mathbf{r}_{\mathsf{head}}, y) S_y. \tag{316}$$

The optimization also fixes a K-basis $(\beta_y)_{y \in \{0,1\}^\kappa}$ of E, forms the packed polynomial

$$g(\mathbf{W}) := \sum_y \beta_y f_y(\mathbf{W}),$$

[14] See Akita PR #460, merged as `4505404b5`, and its transcript-grinding specification.

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

$$g(\mathbf{r}_{\mathsf{tail}}) = \sum_y \beta_y S_y. \tag{317}$$

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

$$\sum_y \chi_y \Delta_y = 0, \qquad \sum_y \beta_y \Delta_y = 0. \tag{318}$$

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 [38, 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. This is an established reduction; Akita uses it here to obtain 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 [80]. 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 [79, Section 5]. Our evaluation is restricted to artifacts that can execute the complete PCS workflow, so there is no Hachi measurement whose security or runtime we normalize against Akita.

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 [57]. 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 [56]. This is the one-product functional identified in (282). The authors credit Quang Dao for identifying the earlier issue. We retain this account of the archived version; our comparisons use the repaired construction.

F.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 the original findings from subsequent repairs and from the security

assumptions and parameter choices that remain different across the measured systems. In particular, 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 the security accounting of our compression maps, in which we price the full input width of each stage.

The findings below concern mismatches between a security argument and the map or parameters instantiated by the released code. We distinguish them 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 C and Section 13.

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 [18,81]. 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.

To see the distinction, 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 [81, 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 price 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.[15]

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 [61]. 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 priced 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 C 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} [61, 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 [61, 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.[16] 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.

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.

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

[15] https://github.com/lattice-arguments/rokoko/issues/94.
[16] https://github.com/lazer-crypto/labrador/issues/1#issuecomment-5718109866.

columns. A fixed map to 32 field elements cannot securely compress binary vectors of arbitrary width merely because their collision bound is one.

The original coefficient projection has a simple algebraic description. Its full 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.

With the protocol context fixed, the width calculation is direct. 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

$$M_{\mathsf{mid}} = 38t_2 = 114$$

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.

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].

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.[17]

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. Orthus has repaired the specific sizing issue recorded here; the old estimates should not be applied to its corrected implementation.

[17] 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.[18]

G Response Bounds and Parameter Estimates

Response bounds have two roles in Akita. The verifier enforces them to bound extracted witnesses, while the honest prover needs to satisfy them without too many retries. This appendix supplies the response estimates used for the second role. Throughout, we use the admission predicates and history-conditional completeness framework of Section 10.1.

We first prove coefficient and squared norm bounds for the actual response distributions, then explain how they certify honest acceptance. We next 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. Changing a bound by any of these methods requires validating the resulting complete schedule, since the next witness and its commitment parameters can change with it.

G.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 G.1 (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

$$V_u := \sum_{a \in \mathcal{Q}} \sum_{v \in \mathrm{supp}(c_a)} |c_{a,v}|^2 s^2_{a,u \ominus v}, \tag{319}$$

where $u \ominus v$ includes the deterministic sign of negacyclic wraparound. Then, for every $t > 0$,

$$\Pr[|z_u| > t] \le 2\exp(-t^2/(2V_u)). \tag{320}$$

If the response has N coefficients and $V_u \le V$ for all u, then

$$\Pr[\|\mathbf{z}\|_\infty > t] \le 2N\exp(-t^2/(2V)). \tag{321}$$

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 (319). Lemma 3.3 gives (320); a union bound over the N coordinates gives (321). The uniform value of V follows by replacing every selected source coefficient by σ_∞ and summing the challenge squared norms.

[18] https://github.com/osdnk/pikku/issues/2; https://github.com/osdnk/pikku/pull/3; https://github.com/osdnk/pikku/pull/4.

Lemma G.2 (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

$$\mathbb{E}[e^{\lambda X}] \le \left(1 + \frac{\nu}{d}(\cosh \lambda - 1)\right)^{n_1} \left(1 + \frac{\nu}{d}(\cosh(2\lambda) - 1)\right)^{n_2}. \tag{322}$$

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 (322).

Theorem G.3 (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

$$M_a(\lambda) := 1 + \frac{\nu_{\mathsf{oh}}}{d_c}\big(\cosh(a\lambda) - 1\big). \tag{323}$$

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

$$\Pr[|z_u| \ge t] \le 2 \inf_{\lambda > 0} \exp\big(W_u\big(n_1 \log M_1(\lambda) + n_2 \log M_2(\lambda)\big) - \lambda t\big). \tag{324}$$

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

$$\|\mathbf{z}\|_\infty \le W \min\{2\nu_{\mathsf{oh}}, n_1 + 2n_2\} \tag{325}$$

is valid simultaneously, so the schedule may take the smaller of the deterministic bound and any threshold certified by (324).

Proof. For one fixed output coefficient, the challenge support samples a population with at most ν_{oh} active unit entries. Lemma G.2 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 (324). 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 (325).

Squared-Norm and Coefficient Bounds for Filtered Challenges A filter changes the challenge distribution. The following bounds use that accepted distribution directly; they do not assume that its coefficient signs remain independent. We begin with the expected response squared norm.

Lemma G.4 (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*

$$\nu_2(\mathcal{D}) := \lambda_{\max}\big(\mathbb{E}_{c\leftarrow\mathcal{D}}[\mathsf{Mul}(c)^{\mathsf{T}}\mathsf{Mul}(c)]\big)\,. \tag{326}$$

Then every fixed source vector $\mathbf{s}$ satisfies

$$\mathbb{E}_{c\leftarrow\mathcal{D}}[\|c*\mathbf{s}\|_2^2] \le \nu_2(\mathcal{D})\|\mathbf{s}\|_2^2. \tag{327}$$

If the challenges c_a are independent and centered, then

$$\mathbb{E}\left[\left\|\sum_{a\in\mathcal{Q}} c_a * \mathbf{s}_a\right\|_2^2\right] \le \sum_{a\in\mathcal{Q}} \nu_2(\mathcal{D}_a)\|\mathbf{s}_a\|_2^2. \tag{328}$$

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 G.5 (Certified squared norm under a symmetric fixed filter). *Let $\mathcal{C}_0$ be a fixed-weight signed sparse shell of challenge squared norm κ_2^2. Let $\mathcal{C}^{\mathsf{acc}} \subseteq \mathcal{C}_0$ be a nonempty accepted family closed under negation, of fraction at least $\underline{\alpha} > 0$, 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 raw signed sparse shell, 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 G.4. 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 lemma requires only negation symmetry of the accepted family; it does not require independent signs within an accepted challenge. The coefficientwise and squared-norm conditions are combined through Corollary G.8. For example, a deterministic coefficient envelope can certify encodability while this lemma certifies the additional Euclidean bound.

Lemma G.6 (Coefficient tails under symmetric fixed 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 shell 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 raw shell, 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. A uniform coefficient envelope B_a gives the simpler choices $m_{a,u} = K_{1,a} B_a$ and $E_{a,u}^{\mathsf{loc}} = d_a B_a^2$. The proof uses independent accepted challenges, not independent coefficient signs within each filtered challenge. All source envelopes must hold after fixing the preceding transcript.

An operator filter does not automatically preserve the unfiltered dense or one-hot tails. To use either bound for a filtered family, one must prove that its accepted distribution retains the independence, centering, and fixed-weight structure used by that result. Otherwise the schedule must analyze the accepted distribution directly, for example with Lemmas G.5 and G.6, or treat the resulting joint admission probability as modeled.

G.2 From Response Bounds to Completeness

The preceding lemmas 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 G.7 (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,

$$\|\mathbf{s}_{h,r,a}(\tau)\|_\infty \le B_{h,r,a}, \qquad \|\mathbf{s}_{h,r,a}(\tau)\|_{2,\mathrm{coef}}^2 \le E_{h,r,a}.$$

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 $\|\mathbf{z}_{h,r}\|_{2,\mathrm{coef}}^2 \le D_{h,r}$. If the protocol centers coefficients modulo q, this can only decrease their absolute values and squared norm. 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. They need not assume 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 sufficient 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, so the resulting schedules must be closed and validated again.

Deterministic envelopes can be conservative. The next route instead assigns a failure budget to each encoding and squared-norm event, then combines the resulting marginal bounds without assuming that the events are independent.

Corollary G.8 (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. The following gives a correlation-agnostic sufficient condition from separate marginal tails. For each response r, choose an encoding-failure allocation $\varepsilon_{h,r}^{\mathsf{enc}} > 0$. For each Euclidean response, also choose a squared-norm failure allocation $\varepsilon_{h,r}^2 > 0$. Suppose*

$$\sum_{r \in \mathcal{R}_h} \left(\varepsilon_{h,r}^{\mathsf{enc}} + \mathbf{1}_{\{p_{h,r}=2\}} \varepsilon_{h,r}^2\right) \le 1 - p_{\mathsf{acc},h}. \tag{329}$$

For every reachable prefix τ, suppose the source-specific analysis gives

$$\Pr_{\boldsymbol{c}_h}[\neg \mathsf{Enc}_{h,r}(\mathbf{z}_{h,r}) \mid \tau, \boldsymbol{c}_h \neq \perp] \le \varepsilon_{h,r}^{\mathsf{enc}}. \tag{330}$$

For a dense response satisfying the conditional Rademacher-sign premises of Lemma G.1, 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

$$Z_{h,r} \ge \sqrt{2V_{h,r} \log\left(\frac{2N_{h,r}}{\varepsilon_{h,r}^{\mathsf{enc}}}\right)}. \tag{331}$$

The one-hot tail of (324) 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

$$S_{\max,h,r} \ge \frac{\mu_{h,r}}{\varepsilon_{h,r}^2}. \tag{332}$$

Then

$$q_h^{\mathsf{rsp}}(\tau) \le 1 - p_{\mathsf{acc},h}, \qquad q_h(\tau) \le 1 - a_h p_{\mathsf{acc},h} \tag{333}$$

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

$$(1 - a_h p_{\mathsf{acc},h})^{N_{\mathsf{try},h}} \tag{334}$$

to (188). No independence between different responses at the same level is required.

Proof. Equation (330) controls the canonical digit decomposition required on both norm routes. For a dense source it follows from Lemma G.1 and (331); for a canonical one-hot root it follows from Theorem G.3. Markov's inequality and (332) control the additional Euclidean event. The union bound and (329) give the pointwise conditional response bound. Equation (182) gives (333). Apply Corollary 10.10.

The corollary makes no independence assumption between encodability and response squared norm. It is useful when those two events are controlled by separate certified bounds. A response model may instead propose a bound on the joint event in Definition 10.5 directly, as in Appendix G.3.

G.3 A Response Model for Parameter Selection

The preceding results certify response bounds under explicit deterministic or distributional premises. The planner may instead use a model to propose bounds and estimate honest retry cost, as discussed in Section 12.3. We record its moment calculations and acceptance parameters here. Their choice does not establish the history-conditional completeness premise for the actual integer responses; that premise is stated separately below.

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

$$\sum_{r \in \mathcal{R}_h} \left(1 - p_{h,r}^{\mathsf{rsp}}\right) \le 1 - p_{\mathsf{acc},h}. \tag{335}$$

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 G.8. 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, not verifier challenges. The schedule separately 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 raw shell and let $L_{h,u}$ be its raw-draw allowance. The planner uses the certified challenge-construction success probability

$$\underline{a}_{h,u} := 1 - (1 - \underline{\alpha}_{h,u})^{L_{h,u}}, \qquad \underline{a}_h := \prod_{u \in U_h} \underline{a}_{h,u} \le a_h. \tag{336}$$

An unfiltered coordinate has $\underline{\alpha}_{h,u} = 1$. Thus the modeled response target and the certified sampler bound combine into complete per-attempt success at least $\underline{a}_h p_{\mathsf{acc},h}$. 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

$$\mathsf{RespPart} := \{\mathsf{z}, \mathsf{e}, \mathsf{t}, \mathsf{r}, \mathsf{cmp}\}.$$

We record the moment estimates separately for each component as

$$\mathsf{M} = \left(d_{\mathsf{pack}}; (\mu_\chi, v_\chi^{\mathsf{full}}, v_\chi^{\mathsf{local}})_{\chi \in \mathsf{RespPart}}\right). \tag{337}$$

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. In particular, the packing update below uses a zero-cross-moment approximation for the local statistic. The modeled total squared norm is $\mu(\mathsf{M}) := \sum_\chi \mu_\chi$. Every stored coordinate 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 deterministic initialization is

$$\mu_{\mathsf{z}} = L \sum_{j=0}^{\delta-1} B_{\mathsf{dig},j}^2, \qquad v_{\mathsf{z}}^{\mathsf{full}} = v_{\mathsf{z}}^{\mathsf{local}} = \max_j B_{\mathsf{dig},j}^2. \tag{338}$$

For a unit one-hot source, the canonical layout determines the number H of unit coefficients after packing exactly, giving

$$\mu_{\mathsf{z}} = H, \qquad v_{\mathsf{z}}^{\mathsf{full}} = v_{\mathsf{z}}^{\mathsf{local}} = 1. \tag{339}$$

Thus neither root state assumes a distribution on the incoming witness. The corresponding coefficient tails are proved in Lemma G.1 and Theorem G.3.

For a uniformly distributed centered base-b digit, define

$$m_2^{\mathsf{unif}}(b) := \frac{1}{b} \sum_{a=-b/2}^{b/2-1} a^2 = \frac{b^2+2}{12}. \tag{340}$$

When a public setup value is modeled as uniform in the field, its digit component sums (340) 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

$$\mathsf{Peak}^{\star}_{N_{\mathsf{src}}}(\mathsf{M}) := \max \left\{ \sum_{\chi \in \mathsf{RespPart}} n_\chi v_\chi^\star : 0 \le n_\chi \le \frac{\mu_\chi}{v_\chi^\star}, \ \sum_{\chi} n_\chi \le N_{\mathsf{src}} \right\}, \tag{341}$$

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 definition prevents disjoint witness components from each being charged 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

$$\widehat{\mu}_{\mathsf{resp}} := \kappa_2^2 \mu(\mathsf{M}), \tag{342}$$

$$\widehat{v}_{\mathsf{resp}} := \frac{\kappa_2^2}{d} \mathsf{Peak}^{\star}_{dB_{\mathsf{col}}}(\mathsf{M}), \tag{343}$$

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

$$\mathsf{GDig}(\sigma, b) := \mathbb{E}_{X \leftarrow \mathcal{N}(0,\sigma^2)} \left[\mathrm{cent}_b(\lfloor X \rceil)^2 \right]. \tag{344}$$

For response base b and digit depth δ_f, the recursive z component is modeled by

$$\mu_{\mathsf{z}}' := N \sum_{j=0}^{\delta_f - 1} \mathsf{GDig}\left(\sqrt{\widehat{\mu}_{\mathsf{resp}}/N}/b^j, b \right), \tag{345}$$

$$v_{\mathsf{z}}' := \max_{0 \le j < \delta_f} \mathsf{GDig}\left(\sqrt{\widehat{v}_{\mathsf{resp}}}/b^j, b \right). \tag{346}$$

Equations (345) and (346) are the Gaussian-inspired step of the model. They are not used for the rigorous root bounds above.

The e, t, and r components use the uniform digit moment of (340) 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 (46) has integer-lift squared norm

$$\|\mathbf{c}\|_2^2 = \sum_t (a_{t,0}^2 + b_{t,0}^2) + 2\sum_t \sum_{j=1}^{k-1} (a_{t,j}^2 + b_{t,j}^2). \tag{347}$$

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 total squared norm and to its full-ring averaged statistic. Its local approximation gives the following update:

$$\mu_\chi^{\mathsf{pack}} = \frac{2k-1}{k}\mu_\chi, \qquad v_\chi^{\mathsf{full,pack}} = \frac{2k-1}{k} v_\chi^{\mathsf{full}}, \qquad v_\chi^{\mathsf{local,pack}} := \begin{cases} v_\chi^{\mathsf{local}}, & k = 1, \\ 2v_\chi^{\mathsf{local}}, & k > 1. \end{cases} \tag{348}$$

This is the implemented model update, not a bound on every packed coordinate. For an overlapping coefficient pair A, B, the exact second moments are

$$\mathbb{E}[(A \pm B)^2] = \mathbb{E}[A^2] + \mathbb{E}[B^2] \pm 2\mathbb{E}[AB]. \tag{349}$$

If both input second moments are at most v, zero raw cross moment gives the local bound $2v$ used by the model. Exchangeability alone does not imply this condition, and independence is insufficient for nonzero-mean inputs. For example, 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 $4v$ instead. A certified coordinate-wise propagation must retain these cross moments or use that bound; the implemented update uses the zero-cross-moment approximation. Applying the upward-rounding map to the five packed components produces the successor model state. Its use to select bounds remains subject to the history-conditional admission premise (184).

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

$$\overline{\mu}_{h,r} := \rho_{\mathsf{model},h}\widehat{\mu}_{h,r}, \tag{350}$$

$$\widehat{v}_{h,r} := \rho_{\mathsf{model},h} \max\{\widehat{\mu}_{h,r}/N, \widehat{v}_{\mathsf{resp},h,r}\}. \tag{351}$$

For every reachable transcript prefix, the model replaces the response by a centered jointly Gaussian surrogate $\mathbf{X}_{h,r}$ satisfying

$$\mathbb{E}[\|\mathbf{X}_{h,r}\|_2^2 \mid \mathsf{tr}_{<h}, \boldsymbol{c}_h \neq \perp] \le \overline{\mu}_{h,r}, \qquad \max_u \mathrm{Var}[X_{h,r,u} \mid \mathsf{tr}_{<h}, \boldsymbol{c}_h \neq \perp] \le \widehat{v}_{h,r}. \tag{352}$$

The coefficient model therefore uses both total and peak moments. 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

$$p_{h,r}^{\mathsf{box}}(\delta) := \left(2\Phi\left(\frac{T_{h,r}(\delta)}{\sqrt{\widehat{v}_{h,r}}}\right) - 1\right)^N. \tag{353}$$

If $\widehat{v}_{h,r} = 0$, define $p_{h,r}^{\mathsf{box}} = 1$ for every nonnegative threshold: the Gaussian surrogate is identically zero. At the terminal, no response-digit witness is created. If the implementation imposes an encoding limit, choose $T_{h,r}(\delta)$ so that every vector in $[-T_{h,r}(\delta), T_{h,r}(\delta)]^N$ fits the selected clear encoding. The box event is then a sufficient condition 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, so the box condition is vacuous. Repeated application of the Gaussian correlation inequality [88] to the coordinate slabs of $\mathbf{X}_{h,r}$ gives

$$\Pr[\|\mathbf{X}_{h,r}\|_\infty \le T_{h,r}(\delta) \mid \mathsf{tr}_{<h}, \boldsymbol{c}_h \neq \perp] \ge p_{h,r}^{\mathsf{box}}(\delta). \tag{354}$$

For a squared-norm bound $S > \overline{\mu}_{h,r}$, Markov's inequality gives the ball score

$$p^{\mathsf{ball}}_{h,r}(S) := 1 - \frac{\overline{\mu}_{h,r}}{S}. \tag{355}$$

If $\overline{\mu}_{h,r} = 0$, the surrogate is identically zero; define $p^{\mathsf{ball}}_{h,r}(0) = 1$ 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

$$\begin{aligned} &\Pr\left[\|\mathbf{X}_{h,r}\|_\infty \le T_{h,r}(\delta) \ \wedge \ \|\mathbf{X}_{h,r}\|_2^2 \le S \mid \mathsf{tr}_{<h}, \boldsymbol{c}_h \ne \bot\right] \\ &\qquad \ge p^{\mathsf{G}}_{h,r}(\delta, S) := p^{\mathsf{box}}_{h,r}(\delta) p^{\mathsf{ball}}_{h,r}(S). \end{aligned} \tag{356}$$

No independence between the coordinates, or between the box and ball events, is used in this calculation.

For a target $p^{\mathsf{rsp}}_{h,r}$ and a depth satisfying $p^{\mathsf{box}}_{h,r}(\delta) > p^{\mathsf{rsp}}_{h,r}$, the least bound implied by the surrogate score is

$$S^{\mathsf{G}}_{h,r}(\delta) := \left\lceil \frac{\overline{\mu}_{h,r}}{1 - p^{\mathsf{rsp}}_{h,r}/p^{\mathsf{box}}_{h,r}(\delta)} \right\rceil. \tag{357}$$

The planner enumerates the admissible digit depths and closes the complete downstream cost of each pair $(\delta, S^{\mathsf{G}}_{h,r}(\delta))$. On the coefficient route it instead requires only $p^{\mathsf{box}}_{h,r}(\delta) \ge p^{\mathsf{rsp}}_{h,r}$.

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

$$\Pr[\mathsf{HAdm}_{h,r}(\mathbf{z}_{h,r}) \mid \mathsf{tr}_{<h}, \boldsymbol{c}_h \ne \bot] \ge p^{\mathsf{rsp}}_{h,r} \tag{358}$$

for every reachable prior transcript. Equation (356) 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 G.9 (From per-response model premises to stage completeness). *Suppose* (358) *holds for every response at level h, with response targets satisfying* (335). *Then the level satisfies* (184) *with* $\widehat{\delta}^{\mathsf{rsp}}_h = 1 - p_{\mathsf{acc},h}$ *and* (183) *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 (335) gives the history-conditional one-shot claim; no independence between responses is required. Fresh random-oracle nonces, the bounded-sampler guarantee (336), 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

$$Z_\infty^2 \approx 2\sigma^2 \log\left(\frac{2N_A}{\varepsilon_\infty}\right), \tag{359}$$

whereas the total squared norm concentrates around $N_A\sigma^2$. A corresponding squared-norm bound has the approximate form

$$S_{\max} \approx \sigma^2 \left(N_A + 2\sqrt{N_A \log(1/\varepsilon_2)} + 2\log(1/\varepsilon_2)\right). \tag{360}$$

When N_A dominates $\log(1/\varepsilon_2)$, these estimates give

$$\frac{N_A Z_\infty^2}{S_{\max}} \approx 2\log\left(\frac{2N_A}{\varepsilon_\infty}\right). \tag{361}$$

The box loses a logarithmic factor in squared radius, or its square root in radius. Thus the ℓ_2 guarantee remains useful even when the ℓ_∞ bound uses the same response model: it proves that many coordinates cannot simultaneously approach the bound.

G.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

$$\sum_{i=0}^{N-1} g_{b,\delta}(z_i) \leq N A_{b,\delta}(\theta) + \theta S. \tag{362}$$

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. A finite search over rational θ gives certified upper bounds without a model of the digit distribution.

This bound supplies a certified alternative to a modeled digit squared-norm entry, but it 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, even 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. Any enlarged envelopes must be propagated through the complete schedule and its Module-SIS parameters; this certificate does not change the model-derived classification of a schedule that uses model-derived response bounds.

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 G.5 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.

H 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. The raw observations and their source identities are supplied in the public benchmark artifact [68]. Each reported value is the median of ten measured runs after one warmup. Verification uses one thread. The artifact retains several measurement cohorts; its manifest records their source identities and the result-file digests.

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	Field elements	Values/poly.	Polynomials
2^{27}	2^{22}	2^{22}	1
2^{29}	2^{24}	2^{23}	2
2^{31}	2^{26}	2^{23}	8
2^{33}	2^{28}	2^{23}	32
2^{35}	2^{30}	2^{23}	128

In particular, the 2^{35}-bit row 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.

Query points and reusable public parameters are excluded from communication totals; the different point dimensions still affect the computation.

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
SP1	100-bit target; rate 1/4, 124 queries, 16 work bits, stacking height 21	Product default

I Benchmark Methodology and Detailed Measurements

The measurements in Section 13 use the native interfaces and parameter profiles described in Appendix H. This appendix records their software provenance, measurement procedure, and communication accounting. The lattice results appear in Table 8 and Fig. 10; the hash results appear in Table 9 and Fig. 11. The detailed hash tables below retain both prover thread counts and all input sizes, including separate communication, memory, and reusable preprocessing measurements. The reproduction artifact records their provenance.

I.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 [66].[19]

The standalone Akita implementation is publicly available [65], 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 raw observations supplied in the public benchmark artifact [68]. 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 F.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 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. The implementation revisions for the hash baselines are listed in Table 15; their parameter profiles and timing boundaries are specified in Appendix H.

Table 15: Implementation revisions for the hash-PCS measurements. The linked commits identify the upstream source revisions; security profiles are recorded separately in Table 14.

Implementation	Revision
Plonky2 FRI	`e1c2d354`
Plonky3 FRI and STIR	`3acc8b70`
WHIR (Plonky3)	`3acc8b70`
Binius64 BaseFold	`6a179536`
Flock Ligerito	`b684b125`
WHIR (WorldFnd)	`c03a4a51`
BaseFold (SP1)	`edc07b68`

[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.

I.2 Inputs and Communication Accounting

For a nominal input target $B = 2^b$, coefficient counts are $N = 2^{b-5}$ for the approximately 32-bit fields, $N = 2^{b-6}$ for 64-bit fields, and $N = 2^{b-7}$ for 128-bit fields. 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 capacities are slightly below the nominal 32-bit targets. The opening field need not be the coefficient field. Appendix H 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. 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; OOM denotes an input that did not complete within the memory limit.

I.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 E describes the integration.

I.4 Detailed Hash-Based PCS Measurements

The following tables give the measurements from the hash cohorts described above. Timing columns distinguish commitment and opening with one or eight threads; verification uses one thread throughout. Fields and coefficient counts follow the configurations in Appendix H. Rows marked $^{(1)}$ use batched univariate FRI/STIR once $\log_2 N > 23$.

Table 16: Hash PCS timings. Commit and open times are in seconds; verification is in milliseconds.

Bits committed	Scheme	Commit 1 thr.	Commit 8 thr.	Open 1 thr.	Open 8 thr.	Verify
2^{27}	Akita (p_{32})	0.0391	0.00827	0.51	0.143	6.0
2^{27}	Akita (p_{64})	0.0274	0.00817	0.331	0.0966	3.9
2^{27}	Akita (p_{128})	0.0366	0.00977	0.242	0.0789	4.2
2^{27}	Plonky2 FRI	3.41	0.548	2.79	1.2	14.5
2^{27}	Plonky3 FRI	0.982	0.205	1.15	0.241	7.1
2^{27}	Plonky3 STIR	0.985	0.21	1.63	0.43	5.8
2^{27}	WHIR (Plonky3)	0.0777	0.0205	0.52	0.146	3.4
2^{27}	Binius64 BaseFold	0.0296	0.00664	0.0115	0.00725	0.4
2^{27}	Flock Ligerito	0.0541	0.0094	0.0709	0.0185	1.2
2^{27}	WHIR (WorldFnd)	0.389	0.0673	2.25	0.348	1.0
2^{27}	BaseFold (SP1)	2.13	2.17	3.25	3.25	25.0
2^{29}	Akita (p_{32})	0.141	0.0221	0.703	0.176	7.5
2^{29}	Akita offload (p_{32})	0.14	0.0214	1.07	0.225	6.8
2^{29}	Akita (p_{64})	0.0857	0.0168	0.423	0.113	4.6
2^{29}	Akita offload (p_{64})	0.0842	0.0153	0.826	0.187	4.8
2^{29}	Akita (p_{128})	0.113	0.0216	0.43	0.103	5.1
2^{29}	Plonky2 FRI	13.8	2.37	11.3	4.89	17.9
2^{29}	Plonky3 FRI$^{(1)}$	2.24	0.469	2.31	0.458	7.8
2^{29}	Plonky3 STIR$^{(1)}$	2.24	0.465	3.24	0.861	6.0
2^{29}	WHIR (Plonky3)	0.347	0.0977	2.3	0.715	3.9
2^{29}	Binius64 BaseFold	0.134	0.0406	0.0416	0.0248	0.5
2^{29}	Flock Ligerito	0.211	0.0419	0.338	0.0799	1.3
2^{29}	WHIR (WorldFnd)	1.78	0.31	9.55	1.46	1.1
2^{29}	BaseFold (SP1)	2.98	3	3.37	3.27	25.1
2^{31}	Akita (p_{32})	0.536	0.0748	1.35	0.275	11.7
2^{31}	Akita offload (p_{32})	0.534	0.0736	1.85	0.364	10.2
2^{31}	Akita (p_{64})	0.303	0.0499	0.815	0.191	7.3
2^{31}	Akita offload (p_{64})	0.299	0.0463	1.13	0.24	5.9
2^{31}	Akita (p_{128})	0.41	0.0742	0.85	0.188	9.1
2^{31}	Plonky2 FRI	55.5	9.5	45.7	20.3	19.5
2^{31}	Plonky3 FRI$^{(1)}$	2.61	0.686	2.3	0.46	7.8
2^{31}	Plonky3 STIR$^{(1)}$	2.6	0.687	3.24	0.857	6.1
2^{31}	WHIR (Plonky3)	1.43	0.445	9.34	3.04	4.4
2^{31}	Binius64 BaseFold	0.586	0.23	0.165	0.0923	0.6
2^{31}	Flock Ligerito	0.911	0.165	1.11	0.265	1.2
2^{31}	WHIR (WorldFnd)	7.77	1.37	40.9	6.3	1.2
2^{31}	BaseFold (SP1)	5.53	5.39	3.51	3.32	25.4
2^{33}	Akita (p_{32})	2.08	0.314	2.32	0.449	18.6
2^{33}	Akita offload (p_{32})	2.2	0.367	3.33	0.586	11.9
2^{33}	Akita (p_{64})	1.25	0.32	1.69	0.333	9.9

Table 16 (continued)

Bits committed	Scheme	Commit 1 thr.	Commit 8 thr.	Open 1 thr.	Open 8 thr.	Verify
2^{33}	Akita offload (p_{64})	1.28	0.314	2.47	0.448	7.5
2^{33}	Akita (p_{128})	1.64	0.375	1.98	0.348	12.2
2^{33}	Akita offload (p_{128})	1.66	0.37	2.47	0.441	8.7
2^{33}	Plonky2 FRI	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*
2^{33}	Plonky3 FRI$^{(1)}$	5.16	1.64	2.14	0.462	7.9
2^{33}	Plonky3 STIR$^{(1)}$	5.16	1.65	3.1	0.849	6.2
2^{33}	WHIR (Plonky3)	4.72	1.54	36.9	12.4	4.7
2^{33}	Binius64 BaseFold	2.5	1.16	0.6	0.34	0.7
2^{33}	Flock Ligerito	3.47	0.763	4.47	1.06	1.6
2^{33}	WHIR (WorldFnd)	34.6	5.91	178	27.4	1.3
2^{33}	BaseFold (SP1)	18.8	18.8	3.87	3.71	26.4
2^{35}	Akita (p_{32})	8.43	1.24	5.7	0.946	32.2
2^{35}	Akita offload (p_{32})	10.7	1.75	8.33	1.33	14.1
2^{35}	Akita (p_{64})	4.62	1.3	4.89	0.823	18.0
2^{35}	Akita offload (p_{64})	4.59	1.27	5.46	0.997	10.1
2^{35}	Akita (p_{128})	5.66	1.43	6.35	0.951	22.2
2^{35}	Akita offload (p_{128})	5.74	1.41	6.21	1.01	10.7
2^{35}	Plonky2 FRI	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*
2^{35}	Plonky3 FRI$^{(1)}$	17.2	6.27	2.91	0.687	8.5
2^{35}	Plonky3 STIR$^{(1)}$	17.1	6.27	3.84	1.09	6.9
2^{35}	WHIR (Plonky3)	15.5	5.23	39.1	12.7	6.1
2^{35}	Binius64 BaseFold	10.7	5.43	2.34	1.31	0.9
2^{35}	Flock Ligerito	15.3	3.02	18.6	4.46	1.5
2^{35}	WHIR (WorldFnd)	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*
2^{35}	BaseFold (SP1)	*n/m*	*n/m*	*n/m*	*n/m*	*n/m*

Table 17: Hash PCS communication. Total is the serialized commitment, separately transmitted evaluation, and opening proof; KB denotes 1000 bytes.

Bits committed	Scheme	Commitment (B)	Proof (KB)	Total (KB)
2^{27}	Akita (p_{32})	128	57.7	57.9
2^{27}	Akita (p_{64})	128	59.4	59.6
2^{27}	Akita (p_{128})	128	61.7	61.8
2^{27}	Plonky2 FRI	520	637.2	637.8
2^{27}	Plonky3 FRI	33	410.6	410.6
2^{27}	Plonky3 STIR	34	290.9	290.9
2^{27}	WHIR (Plonky3)	33	211.4	211.4
2^{27}	Binius64 BaseFold	32	308.6	308.7
2^{27}	Flock Ligerito	4137	408.9	413.0
2^{27}	WHIR (WorldFnd)	56	196.6	196.7
2^{27}	BaseFold (SP1)	32	1179.4	1179.5
2^{29}	Akita (p_{32})	128	58.2	58.4
2^{29}	Akita offload (p_{32})	128	62.6	62.7
2^{29}	Akita (p_{64})	128	61.0	61.1
2^{29}	Akita offload (p_{64})	128	67.4	67.5

Table 17 (continued)

Bits committed	Scheme	Commitment (B)	Proof (KB)	Total (KB)
2^{29}	Akita (p_{128})	128	62.5	62.6
2^{29}	Plonky2 FRI	520	780.3	780.8
2^{29}	Plonky3 FRI$^{(1)}$	33	448.4	448.5
2^{29}	Plonky3 STIR$^{(1)}$	34	307.3	307.4
2^{29}	WHIR (Plonky3)	33	251.7	251.7
2^{29}	Binius64 BaseFold	32	390.4	390.4
2^{29}	Flock Ligerito	4137	340.8	345.0
2^{29}	WHIR (WorldFnd)	56	221.6	221.6
2^{29}	BaseFold (SP1)	32	1182.5	1182.6
2^{31}	Akita (p_{32})	128	59.5	59.6
2^{31}	Akita offload (p_{32})	128	62.7	62.9
2^{31}	Akita (p_{64})	128	62.4	62.5
2^{31}	Akita offload (p_{64})	128	64.7	64.8
2^{31}	Akita (p_{128})	128	64.3	64.4
2^{31}	Plonky2 FRI	520	846.0	846.5
2^{31}	Plonky3 FRI$^{(1)}$	33	452.7	452.9
2^{31}	Plonky3 STIR$^{(1)}$	34	312.4	312.6
2^{31}	WHIR (Plonky3)	33	279.9	279.9
2^{31}	Binius64 BaseFold	32	472.8	472.9
2^{31}	Flock Ligerito	4137	494.6	498.8
2^{31}	WHIR (WorldFnd)	56	256.9	256.9
2^{31}	BaseFold (SP1)	32	1194.8	1194.9
2^{33}	Akita (p_{32})	128	60.9	61.0
2^{33}	Akita offload (p_{32})	128	63.3	63.4
2^{33}	Akita (p_{64})	128	62.9	63.1
2^{33}	Akita offload (p_{64})	128	65.0	65.1
2^{33}	Akita (p_{128})	128	64.8	65.0
2^{33}	Akita offload (p_{128})	128	67.1	67.3
2^{33}	Plonky2 FRI	*n/m*	*n/m*	*n/m*
2^{33}	Plonky3 FRI$^{(1)}$	33	470.3	470.9
2^{33}	Plonky3 STIR$^{(1)}$	34	329.9	330.6
2^{33}	WHIR (Plonky3)	33	305.8	305.8
2^{33}	Binius64 BaseFold	32	569.4	569.5
2^{33}	Flock Ligerito	4137	524.8	529.0
2^{33}	WHIR (WorldFnd)	56	281.5	281.5
2^{33}	BaseFold (SP1)	32	1244.0	1244.0
2^{35}	Akita (p_{32})	128	61.0	61.1
2^{35}	Akita offload (p_{32})	128	63.7	63.9
2^{35}	Akita (p_{64})	128	63.0	63.2
2^{35}	Akita offload (p_{64})	128	66.8	67.0
2^{35}	Akita (p_{128})	128	65.0	65.2
2^{35}	Akita offload (p_{128})	128	67.9	68.1
2^{35}	Plonky2 FRI	*n/m*	*n/m*	*n/m*
2^{35}	Plonky3 FRI$^{(1)}$	33	532.7	535.3
2^{35}	Plonky3 STIR$^{(1)}$	34	401.6	404.2
2^{35}	WHIR (Plonky3)	33	410.4	410.4
2^{35}	Binius64 BaseFold	32	666.7	666.8
2^{35}	Flock Ligerito	4137	570.5	574.6

Table 17 (continued)

Bits committed	Scheme	Commitment (B)	Proof (KB)	Total (KB)
2^{35}	WHIR (WorldFnd)	*n/m*	*n/m*	*n/m*
2^{35}	BaseFold (SP1)	*n/m*	*n/m*	*n/m*

Table 18: Hash PCS memory and reusable setup. RSS includes the dense polynomial; state is persistent preprocessing storage.

Bits committed	Scheme	RSS (GiB) 1 thr.	RSS (GiB) 8 thr.	Prep. (s)	State (GiB)
2^{27}	Akita (p_{32})	0.108	0.147	0.0034	0.0020
2^{27}	Akita (p_{64})	0.103	0.133	0.0034	0.0024
2^{27}	Akita (p_{128})	0.118	0.153	0.0059	0.0049
2^{27}	Plonky2 FRI	0.958	0.957	0.0000	0.0000
2^{27}	Plonky3 FRI	1.18	1.17	0.0271	*unknown*
2^{27}	Plonky3 STIR	1.18	1.17	0.0271	*unknown*
2^{27}	WHIR (Plonky3)	0.235	0.24	0.0013	*unknown*
2^{27}	Binius64 BaseFold	0.0918	0.0913	0.0031	*unknown*
2^{27}	Flock Ligerito	0.0896	0.0905	0.0001	0.0000
2^{27}	WHIR (WorldFnd)	0.174	0.181	0.0002	0.0000
2^{27}	BaseFold (SP1)	1.33	1.33	0.0000	*unknown*
2^{29}	Akita (p_{32})	0.222	0.262	0.0035	0.0020
2^{29}	Akita offload (p_{32})	0.232	0.264	0.0091	0.0020
2^{29}	Akita (p_{64})	0.19	0.227	0.0040	0.0029
2^{29}	Akita offload (p_{64})	0.205	0.235	0.0123	0.0039
2^{29}	Akita (p_{128})	0.188	0.223	0.0071	0.0059
2^{29}	Plonky2 FRI	3.82	3.82	0.0000	0.0000
2^{29}	Plonky3 FRI$^{(1)}$	2.41	2.41	0.0617	*unknown*
2^{29}	Plonky3 STIR$^{(1)}$	2.41	2.41	0.0623	*unknown*
2^{29}	WHIR (Plonky3)	0.928	0.936	0.0055	*unknown*
2^{29}	Binius64 BaseFold	0.358	0.357	0.0121	*unknown*
2^{29}	Flock Ligerito	0.364	0.366	0.0002	0.0000
2^{29}	WHIR (WorldFnd)	0.685	0.723	0.0002	0.0000
2^{29}	BaseFold (SP1)	1.56	1.56	0.0000	*unknown*
2^{31}	Akita (p_{32})	0.627	0.663	0.0079	0.0049
2^{31}	Akita offload (p_{32})	0.671	0.696	0.0213	0.0039
2^{31}	Akita (p_{64})	0.421	0.454	0.0076	0.0059
2^{31}	Akita offload (p_{64})	0.487	0.519	0.0307	0.0078
2^{31}	Akita (p_{128})	0.507	0.507	0.0138	0.0117
2^{31}	Plonky2 FRI	15.3	15.3	0.0000	0.0000
2^{31}	Plonky3 FRI$^{(1)}$	2.78	2.78	0.0607	*unknown*
2^{31}	Plonky3 STIR$^{(1)}$	2.79	2.78	0.0603	*unknown*
2^{31}	WHIR (Plonky3)	3.7	3.7	0.0270	*unknown*
2^{31}	Binius64 BaseFold	1.42	1.42	0.0498	*unknown*
2^{31}	Flock Ligerito	1.35	1.35	0.0002	0.0000
2^{31}	WHIR (WorldFnd)	2.73	2.85	0.0002	0.0000
2^{31}	BaseFold (SP1)	2.5	2.5	0.0000	*unknown*
2^{33}	Akita (p_{32})	1.21	1.26	0.0159	0.0107

Table 18 (continued)

Bits committed	Scheme	RSS (GiB) 1 thr.	RSS (GiB) 8 thr.	Prep. (s)	State (GiB)
2^{33}	Akita offload (p_{32})	1.33	1.35	0.0667	0.0156
2^{33}	Akita (p_{64})	1.33	1.38	0.0181	0.0156
2^{33}	Akita offload (p_{64})	1.5	1.54	0.0781	0.0156
2^{33}	Akita (p_{128})	1.41	1.44	0.0340	0.0313
2^{33}	Akita offload (p_{128})	1.5	1.54	0.0991	0.0313
2^{33}	Plonky2 FRI	*n/m*	*n/m*	*n/m*	*n/m*
2^{33}	Plonky3 FRI$^{(1)}$	4.28	4.28	0.0595	*unknown*
2^{33}	Plonky3 STIR$^{(1)}$	4.29	4.28	0.0602	*unknown*
2^{33}	WHIR (Plonky3)	13.7	13.7	0.0616	*unknown*
2^{33}	Binius64 BaseFold	5.67	5.67	0.1971	*unknown*
2^{33}	Flock Ligerito	5.38	5.41	0.0002	0.0000
2^{33}	WHIR (WorldFnd)	10.9	11	0.0002	0.0000
2^{33}	BaseFold (SP1)	6.25	6.25	0.0000	*unknown*
2^{35}	Akita (p_{32})	4.37	4.4	0.0301	0.0215
2^{35}	Akita offload (p_{32})	4.47	4.5	0.1432	0.0313
2^{35}	Akita (p_{64})	4.51	4.61	0.0359	0.0313
2^{35}	Akita offload (p_{64})	4.7	4.77	0.1356	0.0313
2^{35}	Akita (p_{128})	4.75	4.81	0.0653	0.0625
2^{35}	Akita offload (p_{128})	4.91	4.98	0.2066	0.0625
2^{35}	Plonky2 FRI	*n/m*	*n/m*	*n/m*	*n/m*
2^{35}	Plonky3 FRI$^{(1)}$	12.2	12.2	0.0598	*unknown*
2^{35}	Plonky3 STIR$^{(1)}$	12.2	12.2	0.0620	*unknown*
2^{35}	WHIR (Plonky3)	18.9	18.9	0.0624	*unknown*
2^{35}	Binius64 BaseFold	22.7	22.7	0.7512	*unknown*
2^{35}	Flock Ligerito	21.5	21.6	0.0002	0.0000
2^{35}	WHIR (WorldFnd)	*n/m*	*n/m*	*n/m*	*n/m*
2^{35}	BaseFold (SP1)	*n/m*	*n/m*	*n/m*	*n/m*