Air-FRI: Acceleration of the FRI Protocol on the GPU for zkSNARKs

Tanmayi Jandhyala
PhD Student,
Faculty of Engineering,
University of Waterloo

Dr. Guang Gong
Professor,
Faculty of Engineering,
University of Waterloo

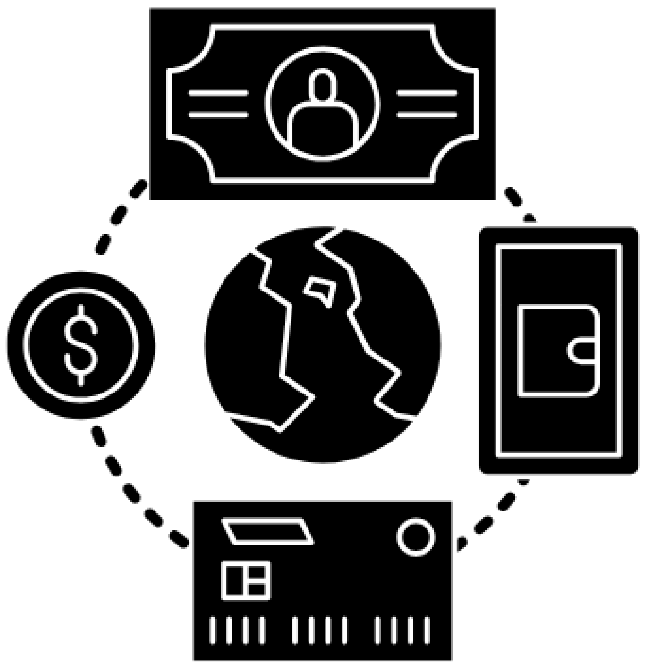
SAC 2025
13th August, 2025

UNIVERSITY OF WATERLOO

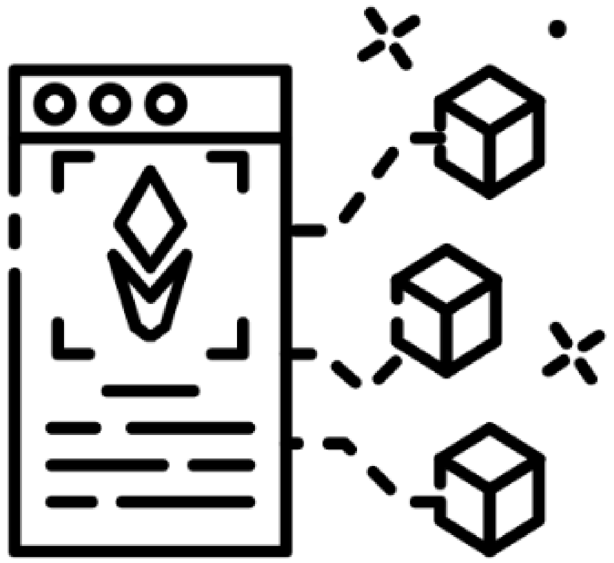
CYBERSECURITY AND PRIVACY INSTITUTE

BTQ

Common applications of Blockchain



Financial Services



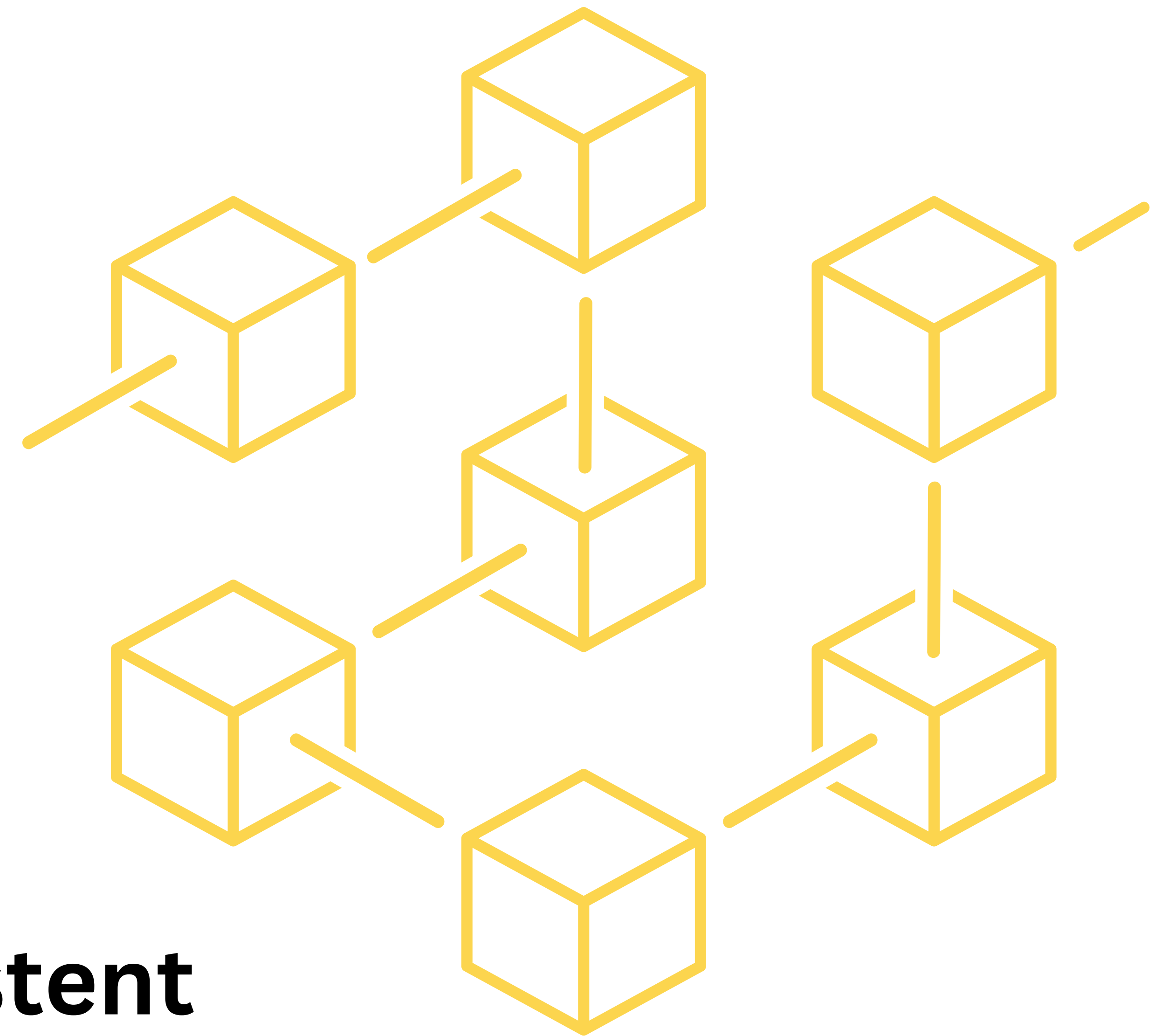
Smart Contracts for trustless agreements

Internet of things (IoT)

Blockchain

decentralized

open ledger



distributed

increased exposure
to traffic or timing analysis

persistent

transaction history
is permanently traceable

Linkability of on-chain transactions to real-world entities

Deanonymization in the Bitcoin P2P Network

Giulia Fanti and Pramod Viswanath

Abstract

Recent attacks on Bitcoin's peer-to-peer (P2P) network demonstrated that its transaction-flooding protocols, which are used to ensure network consistency, may enable user deanonymization—the linkage of a user's IP address with her pseudonym in the Bitcoin network. In 2015, the Bitcoin community responded to these attacks by changing the network's flooding mechanism to a different protocol, known as diffusion. However, it is unclear if diffusion actually improves the system's anonymity. In this paper, we model the Bitcoin networking stack and analyze its anonymity properties, both pre- and post-2015. The core problem is one of epidemic source inference over graphs, where the observational model and spreading mechanisms are informed by Bitcoin's implementation; notably, these models have not been studied in the epidemic source detection literature before. We identify and analyze near-optimal source estimators. This analysis suggests that Bitcoin's networking protocols (both pre- and post-2015) offer poor anonymity properties on networks with a regular-tree topology. We confirm this claim in simulation on a 2015 snapshot of the real Bitcoin P2P network topology.

Deanonymization and linkability of cryptocurrency transactions based on network analysis

Alex Biryukov
University of Luxembourg
alex.biryukov@uni.lu

Sergei Tikhomirov
University of Luxembourg
sergey.s.tikhomirov@gmail.com

WIRED SECURITY POLITICS GEAR THE BIG STORY BUSINESS SCIENCE CULTURE IDEAS MERCH SIGN IN

ANDY GREENBERG THE BIG STORY APR 7, 2022 6:00 AM

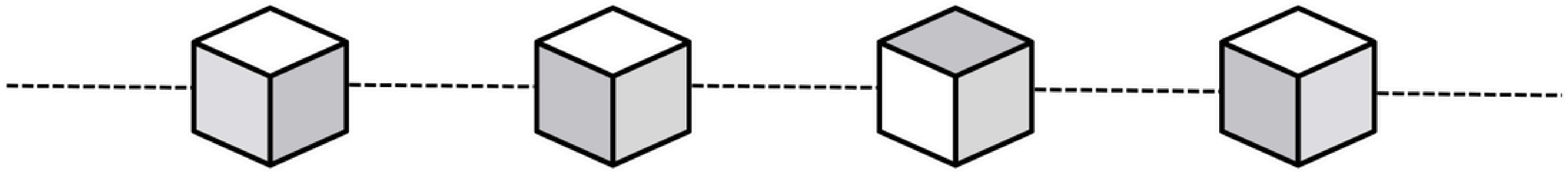
Inside the Bitcoin Bust That Took Down the Web's Biggest Child Abuse Site

They thought their payments were untraceable. They couldn't have been more wrong. The untold story of the case that shredded the myth of Bitcoin's anonymity.

Deanonymisation of Clients in Bitcoin P2P Network

Alex Biryukov Dmitry Khovratovich Ivan Pustogarov
University of Luxembourg
{alex.biryukov, dmitry.khovratovich, ivan.pustogarov}@uni.lu

Ensuring Privacy Through Zero-Knowledge



blockchain nodes vulnerable to de-anonymization

blockchain nodes using zero-knowledge proofs to hide transaction details

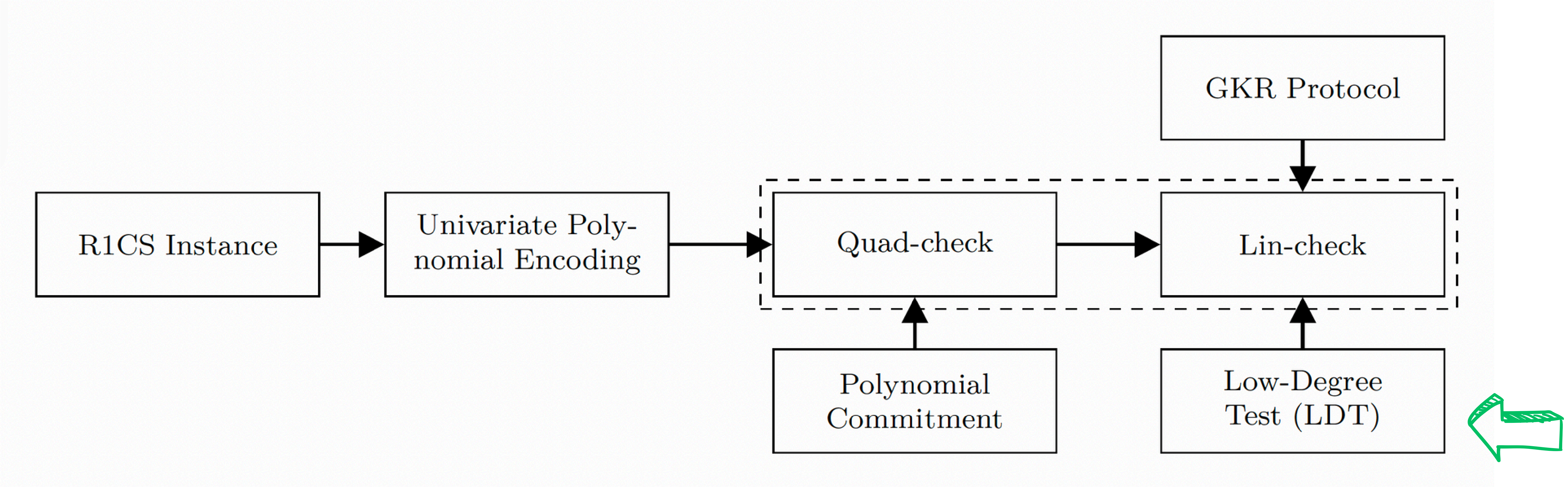
Enabling ZKPs through zkSNARK algorithms

Zero-Knowledge: No additional information is disclosed beyond the validity of the statement.
Succinct: Proofs are short and efficient, even for complex computations.
Non-Interactive: Verification requires only a single proof submission without further interaction.

Specific Applications to digital signatures: Allows users to prove ownership of private keys without revealing them, also enhances security by preventing exposure of sensitive information during the signing process.

Scalable Verification: Efficient proof verification reduces computational load on the blockchain, while supporting the validation of large computations with minimal resource usage.

Overview of Polaris (Fu, Gong ‘22) .



FRI Protocol [Ben+18a]

- Goal is to prove that the degree of a polynomial f(x) is less than d

$$deg(f(x)) < d$$

- The proof process is to iteratively reduce the degree by **half** at each step.
  - Specifically, at each round of the protocol, the prover computes its following codeword
- FRI is an interactive oracle proof where the verifier sends challenges x_i based on the prover's responses.
- The prover commits to the folded polynomial using a Merkle tree.
- In the final round, the verifier performs consistency checks on the reduced polynomial.

FRI Protocol Overview

commit phase

Prover commits to codewords derived recursively from the polynomial

at each round:

- sends a commitment of the codeword to the verifier
- verifier issues a random challenge
- prover constructs its following codeword using a line computation equation

final codeword is sent directly to the verifier.

query phase

verifier extracts
- the commitments (merkle roots) and challenges
- final codeword

and has oracle access to intermediate codewords

verifier checks
- degree of the polynomial interpolated from the final codeword

verifier randomly samples a point in the linear subspace and computes 'indices' around it
- verifier consistency of consecutive codewords at these indices

round-consistency check

verifier iteratively checks for the merkle proofs of committed codewords

computes the line equation for the points selected in the previous phase.

any one check fails, and the proof is rejected

So what's the problem?

- The FRI low-degree test uses large mathematical constraints, which in implementation, would involve complex memory usage and data structures.

- For example, Preon, a zk-SNARK algorithm which is programmed in C, has the below prover and verifier times for the FRI Protocol.

Metric	Prover Time (seconds)	Verifier Time (seconds)
Average	135.83	1.65
Minimum	129.04	1.24
Maximum	212.68	3.12

Table 1: Preon's Prover and Verifier Times for L3 Parameters over 50 Iterations on the CPU

In order for zk-SNARK schemes which use FRI as one of their core components to be applied to real-world privacy-preservation, they need to be efficient in their performance.

We identified performance-intensive parts of the protocol and optimized them for implementation.

Pre-compute field inverses

$$f_{k+1}(L_{k+1}[i]) = \frac{f_k(L_k[2i]) - f_k(L_k[2i+1])}{L_k[2i] - L_k[2i+1]} \cdot (\alpha^{(k)} - L_k[2i]) + f_k(L_k[2i]),$$

can be pre-computed!

L_k : Domain of the current codeword.

$\alpha^{(k)}$: Uniform random challenge from the verifier

$L_k[2i] - L_k[2i+1]$: basis element

Transforming the protcol to non-interactive [COS20]

- The protocol leverages **repeated hashing** of the uniformly sampled element in each round to eliminate direct interaction with the verifier.
- Instead of transmitting intermediate data, the prover uses deterministic hash operations to generate challenges.

$$\alpha^{(1)} = Hash(\alpha^{(0)} \parallel root),$$

$$\alpha^{(k+1)} = Hash(\alpha^{(k)})$$

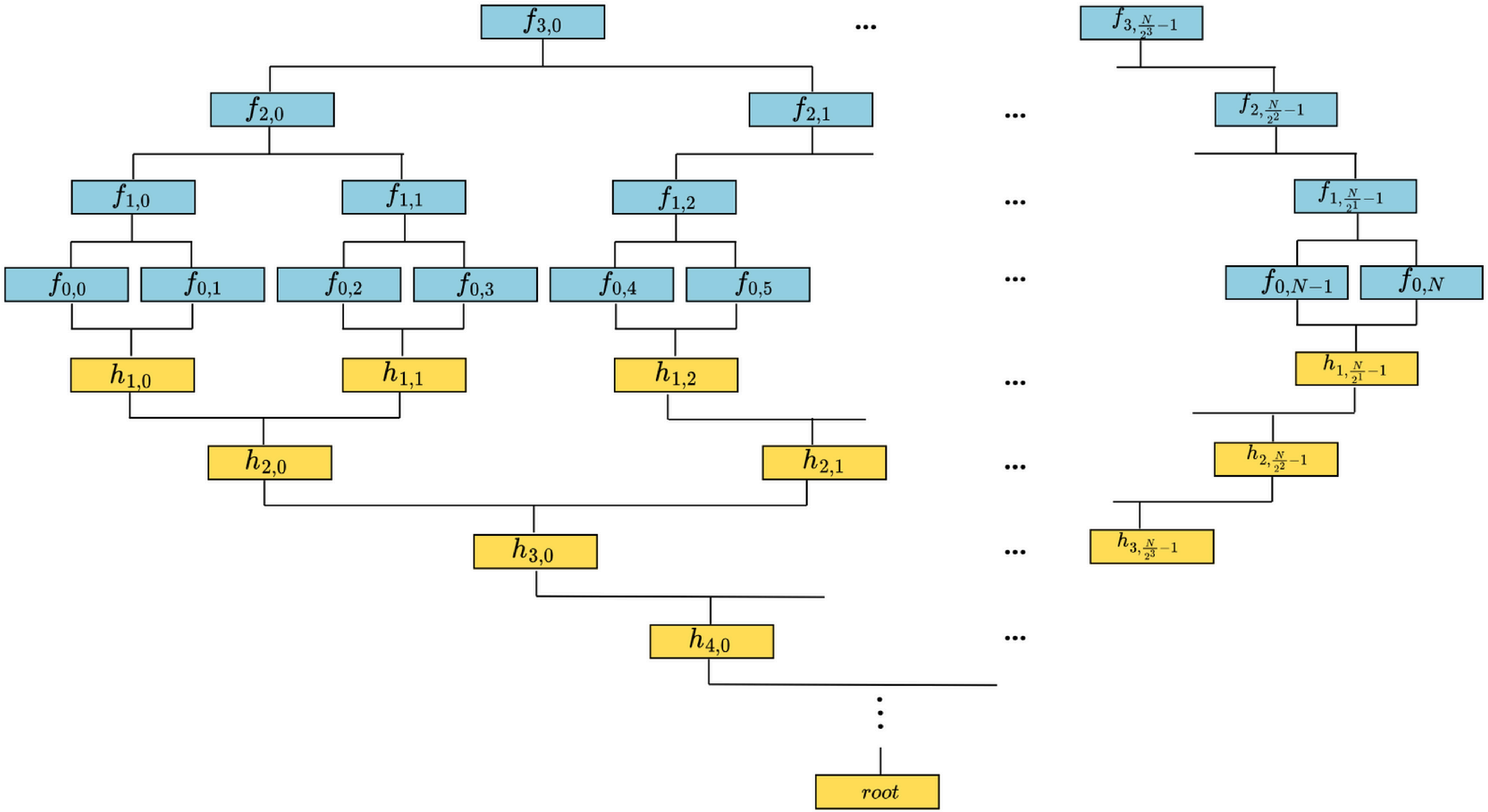
$\alpha^{(k)}$: Challenge for the current round.

$root$: Merkle root of the codeword that is initially computed.

$1 \leq k \leq r$, where r is the number of rounds of the FRI protocol

Parallel computation of codeword elements on the GPU

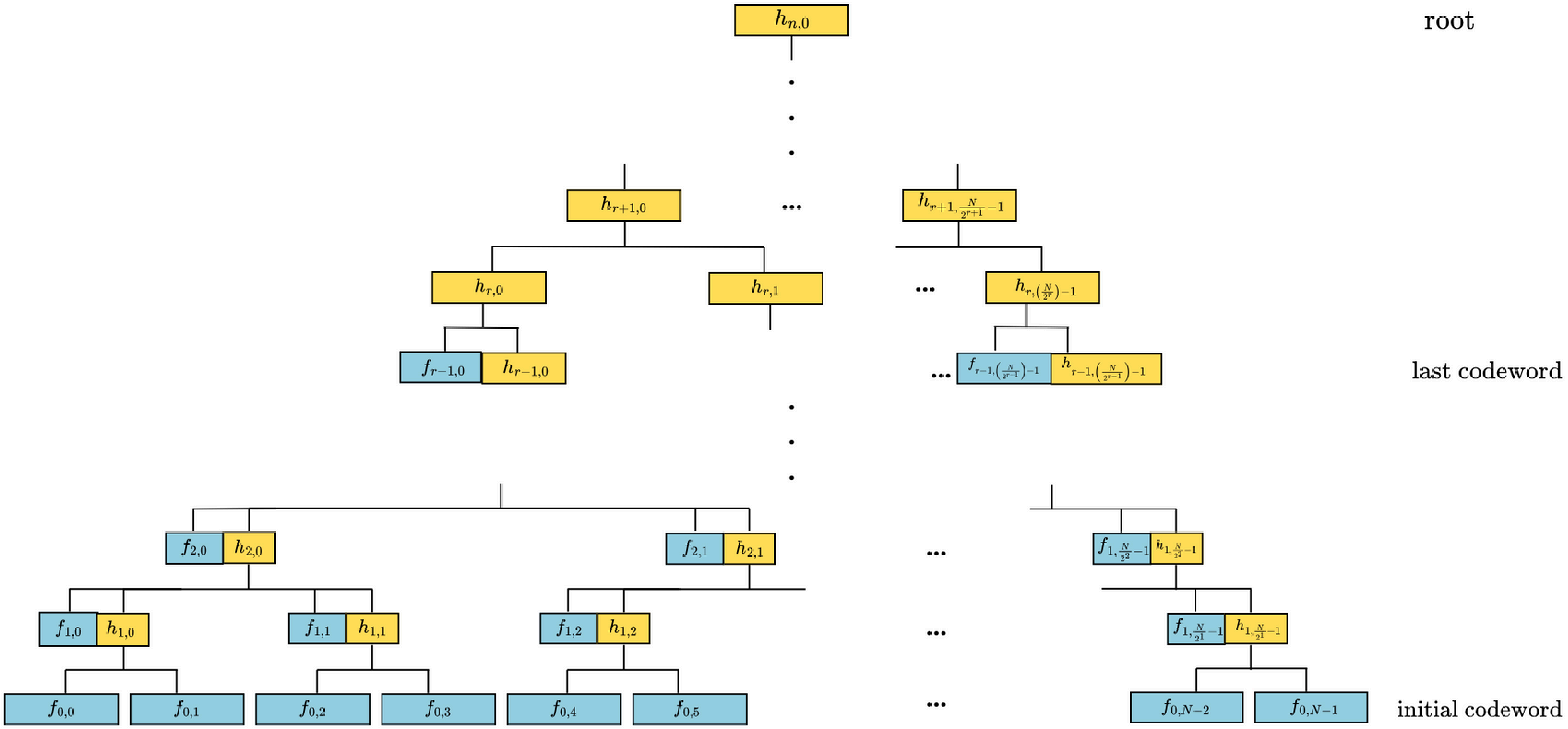
- **Each GPU thread computes elements of the codeword in parallel**



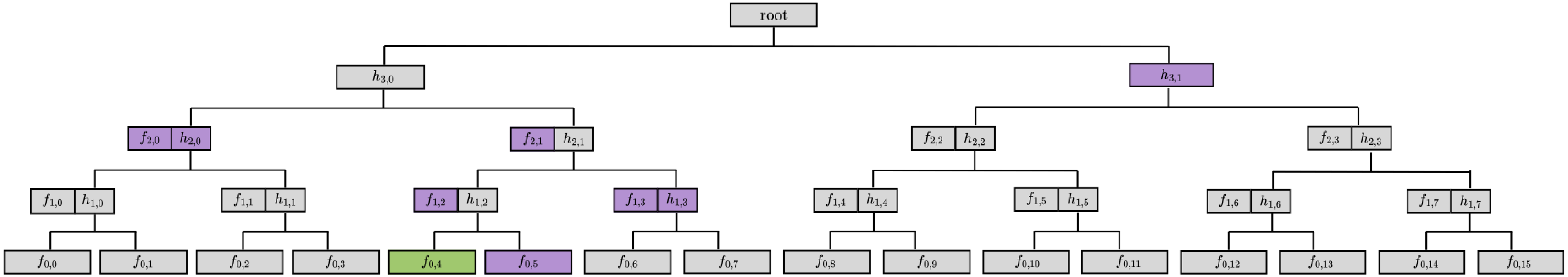
Integrated FRI Merkle Tree

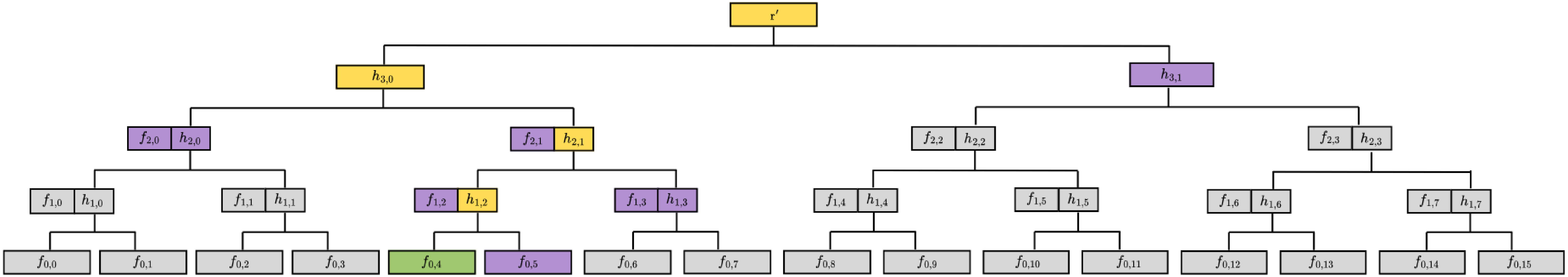
- Combines RS codeword evaluations with a Merkle tree for scalability and security.
- Reduces the prover's overhead while ensuring verifier efficiency in zkSNARKs.

Structure of our specially-constructed tree

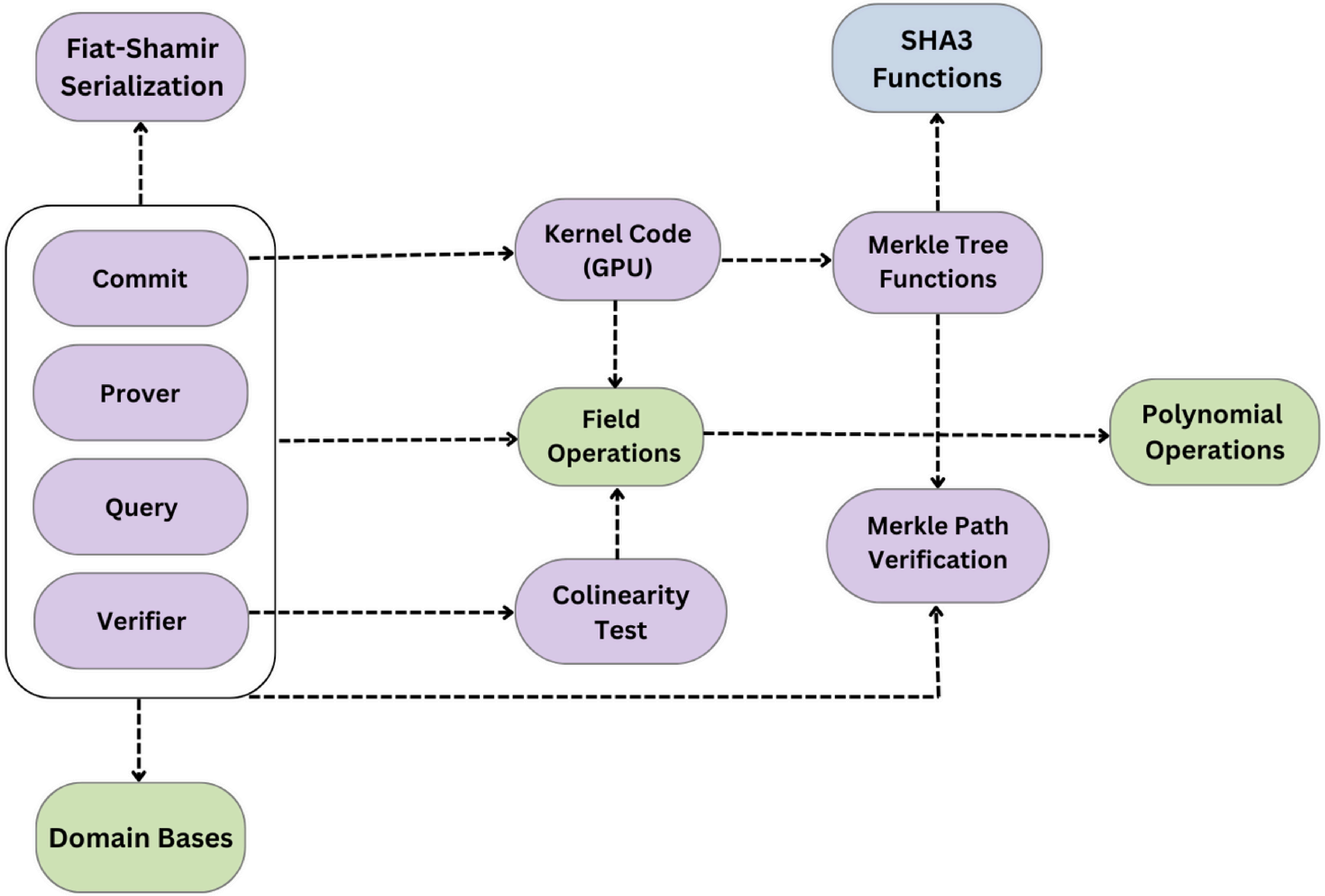


Verification Process:





Components of Air-FRI:



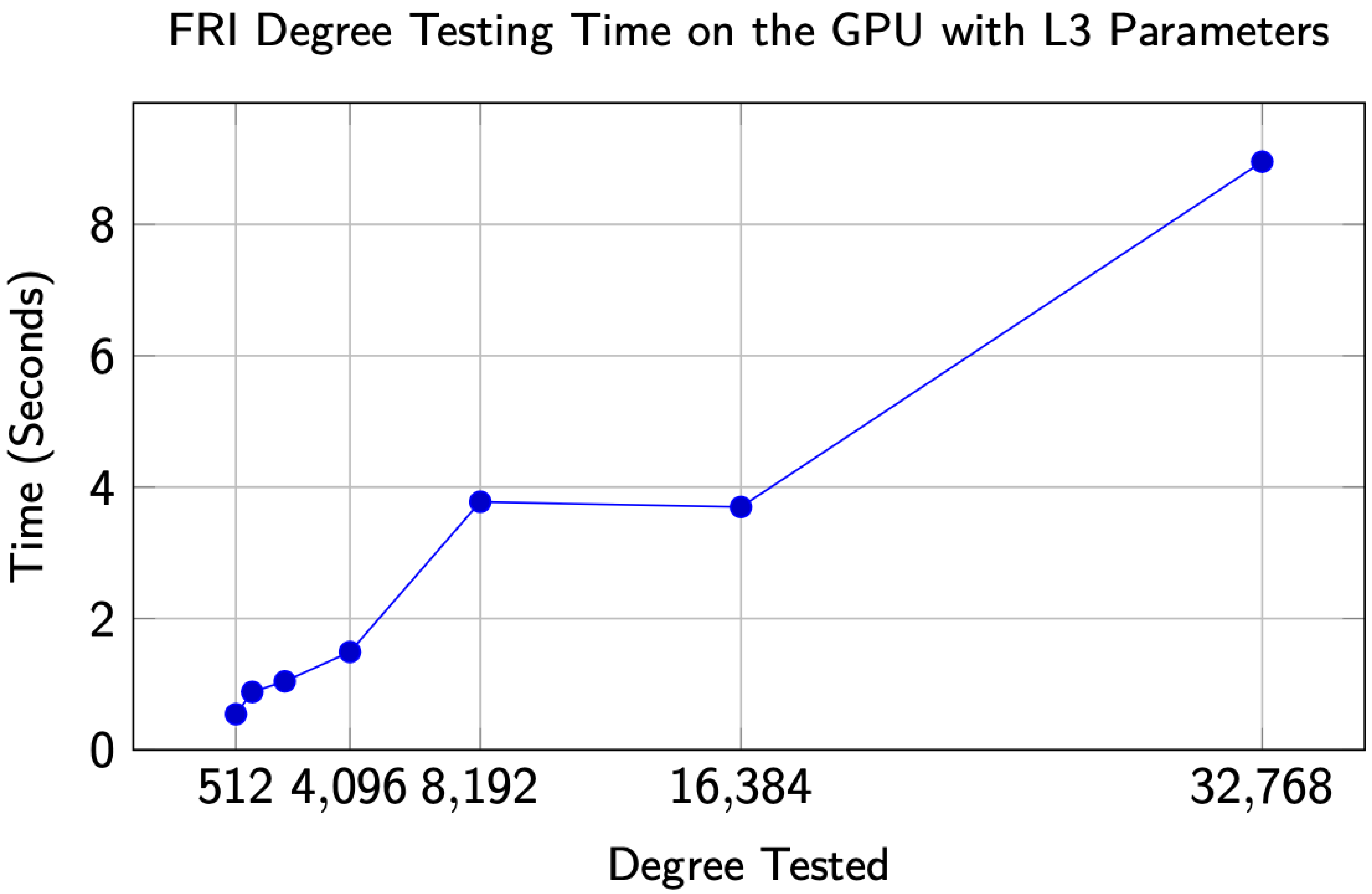
Results

L3 Parameters Overview:

Parameter	Value
Finite Field (L3)	2^{256}
Degree Tested	2^{12}
Expansion Factor	32
Least Codeword Length	2^5 (32)
Initial Codeword Length	2^{17} (131072)
Number of Co-linearity Tests	14
Number of Codeword Reductions (Rounds)	12
Merkle Tree Height	17
Hash Function Input Output	$1024 \rightarrow 256$

S.No.	Category	Iterations	Average (s)	Max (s)	Min (s)
1	C Code - No Optimizations	100	22.52	24.49	21.28
2	C Code - Precomputed Inverses	100	17.95	18.29	17.67
3	C Code - Verifiable FRI Merkle Tree	100	13.47	14.39	12.97
4	GPU Code - Parallel Line Computation	100	9.36	10.23	9.14
5	**GPU Code - Parallel Line Computation & Verifiable Merkle Tree**	**100**	**1.49**	**1.63**	**1.46**

Performance comparison on CPU vs GPU for L3


FRI Degree Testing Time on the GPU with L3 Parameters
Time (Seconds)
0
2
4
6
8
512
4,096
8,192
16,384
32,768
Degree Tested

Degree	CPU Time (s)	GPU Time (s)	Speedup Factor (%)
512	2.80	0.54	80.59%
1024	5.48	0.88	83.90%
2048	10.86	1.04	90.39%
4096	**22.43**	**1.49**	**93.35%**
8192	45.16	3.78	91.64%
16384	104.37	3.70	96.46%
32768	205.37	8.95	95.64%

Speedup Factor (%) of GPU over CPU for Various Degrees (L3)

Metric	Prover Time (Preon)(s)	Verifier Time (Preon)(s)	Prover Time (GPU)(s)	Verifier Time (GPU)(s)
Minimum	126.926	0.803	0.669	0.165
Maximum	139.683	1.167	0.674	0.167
Average	129.248	0.875	0.670	0.166

Comparison of Prover and Verifier Times between Preon [Che+23] and Air-FRI for Degree 2^{12}

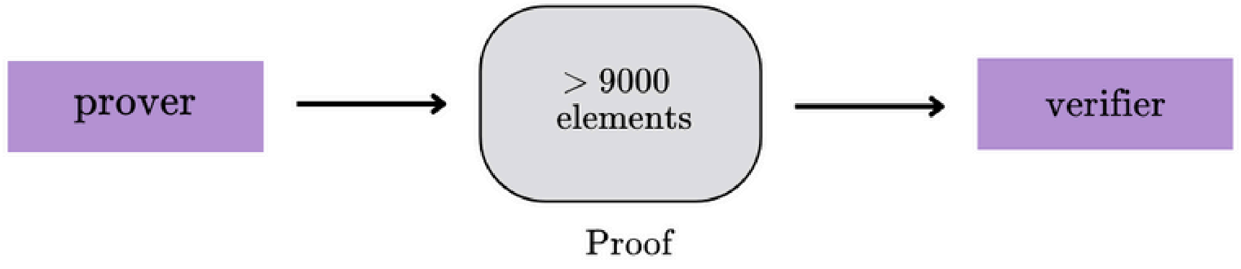
Prover and verifier times measured over 10 iterations on the GPU for L5 Parameters ($\mathbb{F}_{2^{320}}$).

Metric	Prover Time (s)	Verifier Time (s)
Minimum	13.85859	1.12372
Maximum	13.88316	1.12699
Average	13.87129	1.12488

- This shows significant promise in realizing our implementation in post-quantum secure algorithms which require such large constraints and strong security levels.

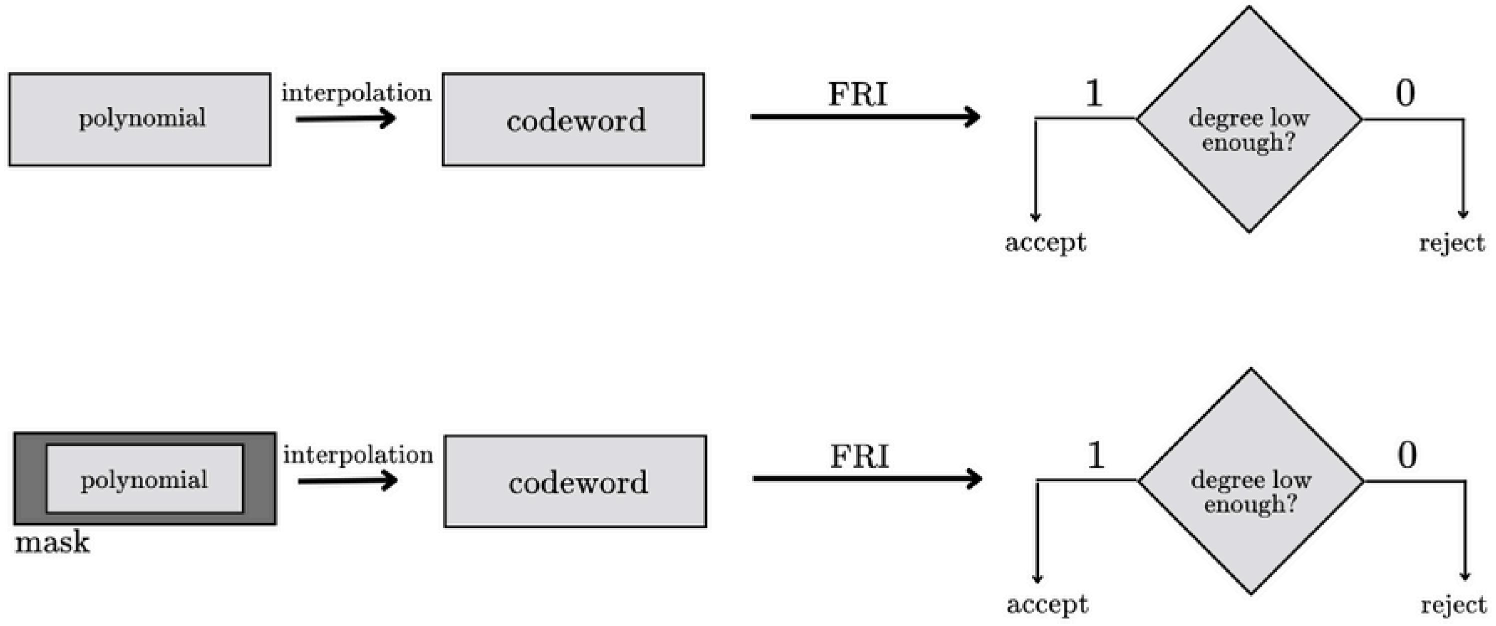
Future Work

1. Further Reduction of Proof Size



2. Faster FFT Implementation: Integrating a more efficient FFT algorithm using special bases and subfield structures could improve performance.

3. Zero-Knowledge Integration



Future Work Cont.

4. Establish soundness guarantees

5. Measure energy consumption of running this protocol on GPUs.

Open to questions!