Sampling for Proximity and Availability

Alex Evans
aevans@baincapital.com

Nicolas Mohnblatt
nmohnblatt@baincapital.com

Guillermo Angeris
gangeris@baincapital.com

November 2024

1 Note

This article is a PDF version of the blog post found at

`https://baincapitalcrypto.com/sampling-for-proximity-and-availability/`

2 Introduction

Blockchains allow nodes to verify the validity of the chain without trusting validators, miners, or block producers. In early designs, this was achieved by having each node download and re-execute every transaction. The cost of that approach motivated the adoption of fraud proofs and validity proofs: a node can check that the state transition was computed correctly without repeating the full computation itself.

These proofs do not, by themselves, guarantee that the data needed to compute the latest state is available. A malicious block producer may publish a valid proof while withholding the data required to reconstruct the underlying state. In that case, a node may know that every state change was valid, but still be unable to learn what the state actually is. This kind of attack can prevent users from composing valid transactions, such as withdrawals from a pool, and can also prevent other nodes from constructing proofs of their own.

The direct way to ensure that the data can be downloaded is to download it. Most current blockchains follow this approach. As usage grows, however, requiring every node to download all block data strains network bandwidth and makes it harder for resource-constrained nodes to participate.

Data availability sampling. Data availability sampling (DAS), introduced in [ASB18], gives light nodes a probabilistic way to verify availability without downloading an entire block. The high-level idea is simple. First, the block data is encoded with an error correcting code. Second, many light nodes sample small parts of the encoded data. If enough independently sampled symbols have been collected across the network, then the data is available: it can either be downloaded directly or reconstructed from those samples.

For this to work, a DAS protocol needs two basic properties. First, sampled data must be known to come from a correctly computed encoding. Second, enough nodes must be sampling

so that their samples collectively contain enough symbols to reconstruct the original data. Production-oriented DAS protocols usually handle the encoding-correctness requirement with fraud proofs or KZG commitments.

Fraud proofs are efficient for the encoder, but they introduce latency and a trust assumption: each light node must be connected to at least one honest full node. KZG commitments avoid that assumption, but they require a trusted setup and are expensive to compute compared with the encoding itself.

Hash-based proofs. Recent work on DAS foundations and FRIDA points to a third path [HSW23, HSW24]. Instead of using KZG commitments, a protocol can adapt efficient hash-based proof systems, such as Ligero [Ame+17] and FRI [Ben+18], to show that the encoding was correctly computed. These constructions have appealing properties: efficient provers, weaker cryptographic assumptions, and no trusted setup. Their main drawback is that the commitments can be large. Since every light node must download the commitment, this overhead can dominate the cost and make the construction impractical.

This article. The point of this article is that the overhead can be reduced by merging two tasks that are usually treated separately. If a light node already makes enough random queries during DAS, those queries can also be used to check proximity to a valid encoding. Sampling for proximity and sampling for availability can be the same action.

3 Sampling for Proximity and Availability

The FRIDA construction. In FRIDA [HSW24], the prover starts with data encoded by a Reed–Solomon code. It then commits to the entries of the encoded vector with a Merkle tree. The prover also runs the FRI proving algorithm to produce a non-interactive proof that the committed vector is within unique decoding distance of a Reed–Solomon codeword. This is a proof of proximity.

The proof contains L queries to the original vector. The parameter L is chosen to reach a target security level. These queries are shared with every light node as part of the commitment to the data.

Each light node downloads the commitment, verifies the non-interactive proof, and then makes Q additional interactive random queries. Those additional queries serve as DAS samples. A key observation in FRIDA is that FRI verification can also be run on these additional queries. This shows that the received symbols are exactly symbols of the unique nearby codeword whose existence was established by the initial proof.

Equivalently, the common non-interactive proof, with its L queries, convinces every light node that the prover committed to a vector close to a unique Reed–Solomon codeword. The Q extra queries are sampled independently across nodes and, after FRI verification, become certified symbols of that same codeword. Enough such certified symbols can be assembled to decode the data.

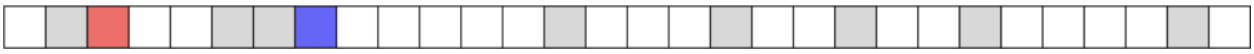

Figure 1: Samples of a codeword in vanilla FRIDA. Grey squares denote non-interactive samples common to both nodes ($L = 9$). Red squares denote the $Q = 1$ sample from the first light node, while blue squares denote the $Q = 1$ sample from the second light node.

The example in Figure 1 shows $N = 2$ light nodes, with $Q = 1$ query each and $L = 9$ non-interactive queries in the commitment. Each light node downloads $L + Q = 10$ samples, or one third of the encoded data. But at most two of these samples are unique across the two nodes, because the L proof queries are identical for both nodes. If the code rate is $1/2$ and the original message has size 15, then the network needs at least 15 samples to decode the message. The two light nodes cannot reach that threshold unless additional nodes sample or these nodes issue more queries.

Leveraging interaction. A small amendment removes this repeated overhead. Instead of sending the same non-interactive proof of size L to each light node, each light node can interactively request and verify its own independent set of L samples. In this variant, the separate Q availability samples are no longer needed. The query phase of FRI is run with interactive randomness, which is no less secure for the individual light node.

Figure 2: Samples of a codeword in interactive FRIDA. Red squares denote samples held only by the first light node, blue squares denote samples held only by the second light node, and purple squares denote samples held by both nodes.

The same two-node example now behaves differently. With the parameters above, the nodes sample enough unique data to decode with probability above 99 percent. Purple entries in Figure 2 are redundant samples that both nodes happen to request. The bad event is that the two nodes overlap on at least nine positions, leaving fewer than 15 unique samples. That event has probability roughly 2^{-28} instead of probability 1.

The non-interactive samples in the vanilla construction are therefore pure overhead from the network's perspective. After the first node downloads them, they add no new information that helps reconstruct the original message. Interactive proximity queries, in contrast, are also availability samples.

Efficiency. This change can make a DAS construction much more efficient. For 80 bits of security, the FRIDA parameter choice is $L = 128$. At the same security level, a scheme in which each node makes $128 + 1$ interactive queries requires two orders of magnitude fewer light nodes than a scheme in which each node makes one fresh availability sample after downloading a shared non-interactive proof with $L = 128$ queries.

The reason is not that interaction changes what FRI proves. It changes how the queries are distributed across the network. The same sampled symbols now serve two roles: they verify proximity to the code and they contribute independent information for decoding.

Comparisons. The interactive variant has one important restriction. To match the standard FRI security analysis, light nodes should sample without replacement. In contrast, the modular DAS primitives in [HSW23] and [HSW24] allow more flexibility in how nodes sample. For practical protocols, the efficiency gain appears to be worth this restriction.

4 Future Work

Interactive queries increase the efficiency of FRIDA-based DAS, but they do not remove all overhead. Standard FRI verification uses correlated queries across rounds. Since queries to later FRI oracles are perfectly correlated with queries to the original oracle, those later queries do not appear to give the network additional information that helps decode the original data.

This leaves a natural question. Can we design a protocol in which every proof query both helps prove that the encoding is correct and contributes information that can be used to decode the original data?

5 Acknowledgements

We thank Kobi Gurkan, John Adler, Nashqueue, and Sanaz Taheri for useful discussions about data availability, its requirements, and possible constructions. Any mistakes or silliness are ours.

References

[ASB18] M. Al-Bassam, A. Sonnino, and V. Buterin, "Fraud Proofs: Maximising Light Client Security and Scaling Blockchains with Dishonest Majorities," *CoRR*, 2018, [Online]. Available: http://arxiv.org/abs/1809.09044

[HSW23] M. Hall-Andersen, M. Simkin, and B. Wagner, "Foundations of Data Availability Sampling." [Online]. Available: https://eprint.iacr.org/2023/1079

[HSW24] M. Hall-Andersen, M. Simkin, and B. Wagner, "FRIDA: Data Availability Sampling from FRI," in *Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part VI*, L. Reyzin and D. Stebila, Eds., in Lecture Notes in Computer Science, vol. 14925. Springer, 2024, pp. 289–324. doi: 10.1007/978-3-031-68391-6_9.

[Ame+17] S. Ames, C. Hazay, Y. Ishai, and M. Venkitasubramaniam, "Ligero: Lightweight sublinear arguments without a trusted setup," in *Proceedings of the 2017 acm sigsac conference on computer and communications security*, 2017, pp. 2087–2104.

[Ben+18] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev, "Fast reed-solomon interactive oracle proofs of proximity," in *45th international colloquium on automata, languages, and programming (icalp 2018)*, 2018, pp. 14–11.