Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.073237

ARTICLE

ISTIRDA: An Efficient Data Availability Sampling Scheme for Lightweight Nodes in Blockchain

Jiaxi Wang[1], Wenbo Sun[2], Ziyuan Zhou[1], Shihua Wu[1], Jiang Xu[1] and Shan Ji[3,*]

[1]School of Computer Science, School of Cyber Science and Engineering, Engineering Research Center of Digital Forensics, Ministry of Education, Nanjing University of Information Science and Technology, Nanjing, 210044, China
[2]School of Software, Shandong University, No. 1500, Shunhua Road, High-Tech Industrial Development Zone, Jinan, 250101, China
[3]College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, No. 169, Sheng Tai West Road, Nanjing, 210016, China

*Corresponding Author: Shan Ji. Email: shanji@nuaa.edu.cn

Received: 13 September 2025; Accepted: 12 November 2025; Published: 10 February 2026

ABSTRACT: Lightweight nodes are crucial for blockchain scalability, but verifying the availability of complete block data puts significant strain on bandwidth and latency. Existing data availability sampling (DAS) schemes either require trusted setups or suffer from high communication overhead and low verification efficiency. This paper presents ISTIRDA, a DAS scheme that lets light clients certify availability by sampling small random codeword symbols. Built on ISTIR, an improved Reed–Solomon interactive oracle proof of proximity, ISTIRDA combines adaptive folding with dynamic code rate adjustment to preserve soundness while lowering communication. This paper formalizes opening consistency and prove security with bounded error in the random oracle model, giving polylogarithmic verifier queries and no trusted setup. In a prototype compared with FRIDA under equal soundness, ISTIRDA reduces communication by 40.65% to 80%. For data larger than 16 MB, ISTIRDA verifies faster and the advantage widens; at 128 MB, proofs are about 60% smaller and verification time is roughly 25% shorter, while prover overhead remains modest. In peer-to-peer emulation under injected latency and loss, ISTIRDA reaches confidence more quickly and is less sensitive to packet loss and load. These results indicate that ISTIRDA is a scalable and provably secure DAS scheme suitable for high-throughput, large-block public blockchains, substantially easing bandwidth and latency pressure on lightweight nodes.

KEYWORDS: Blockchain scalability; data availability sampling; lightweight nodes

1 Introduction

Blockchain's decentralization, immutability, transparency, and traceability have driven impact across finance [1–3], supply chains [4–6], healthcare [7–9], and digital identity [10–13], enabling more efficient, secure, and auditable systems [14]. Nevertheless, scalability remains a primary barrier [15]: Ethereum processes about 60 transactions per second [16], whereas conventional payment rails sustain roughly 1700 TPS and peak near 24,000. The proliferation of decentralized applications (dApps) aggravates congestion and confirmation delays [17,18], reinforcing the urgency of scalable designs for broad adoption.

The use of lightweight nodes [19,20] is key to blockchain scalability. By storing only block headers, they cut storage and computation costs, lowering the participation barrier and promoting decentralization [21]. But without full block data, they cannot directly verify transactions and thus rely on a secure data-availability mechanism. We focus on public blockchains, where full and lightweight nodes coexist and decentralization

hinges on the participation of resource-constrained clients. We assume an open, adversarial peer-to-peer setting (e.g., Ethereum-like networks), where lightweight nodes cannot trust arbitrary relays and instead depend on cryptographic data-availability guarantees. This defines our threat model and performance goals, and differentiates our work from permissioned or consortium settings.

Data availability sampling (DAS) [22] is a core cryptographic technique used to resolve the security and performance trade-off in blockchain. This technique allows light nodes to efficiently verify the availability of all data by randomly sampling a small amount of data. This enables light nodes to ensure network security while maintaining low resource consumption.

However, existing DAS designs face various deployment and performance limitations. Hall-Andersen et al. proposed two constructions of DAS schemes [23]. One construction uses vector commitments combined with succinct non-interactive arguments of knowledge (SNARKs), which is sound but computationally expensive and relies on strong cryptographic assumptions. Another, adopted by Ethereum, uses two-dimensional Reed–Solomon (RS) codes [24] with Kate–Zaverucha–Goldberg (KZG) polynomial commitment scheme [25], offering good performance at the expense of a heavy trusted setup. More recently, Hall-Andersen et al. introduced a DAS scheme called FRIDA [26], which is built on FRI [27], a Fast Reed–Solomon interactive oracle proof of proximity (IOPP). FRIDA does not require trusted setup, but still incurs high constant factor communication costs and uses a fixed code rate that does not adapt well to different data scales [28].

To overcome these limitations, we turn to improvements at the IOPP layer. In particular, our work is inspired by the Shift-to-Improve-Rate (STIR) protocol [29]. STIR improves efficiency by reducing the polynomial degree and code rate through recursion. However, its fixed folding ratio and code rate can lead to redundancy or premature shrinking of the query domain. We therefore develop ISTIR, an improved RS IOPP with adaptive folding and dynamic code-rate adjustment. Building on ISTIR, we design ISTIRDA, a DAS scheme tailored for high-frequency and large-data settings. The main contributions of this work are summarized as follows:

- This paper presents a DAS that couples recursive RS degree reduction with dynamic code-rate adjustment, restructuring the IOPP to cut communication and accelerate verification—effects most pronounced at large data scales, easing lightweight-node bandwidth pressure.
- This paper proposes an adaptive folding method that selects the folding ratio and adjusts the code rate of each round based on the data size and security requirements, thereby maintaining efficiency and avoiding redundant communication. After 16 MB, ISTIRDA is faster than FRIDA in verification speed, and the gap will become larger and larger as the data size increases.
- This paper provides formal security under standard assumptions and a prototype evaluation: at D = 128 MB, the proof size of ISTIRDA is 60% smaller and the verification time 25% shorter than FRIDA, with the query complexity, the verification time, and proof size reported.

2 Related Work

2.1 Data Availability and Origin of DAS

Data availability refers to the ability of all participants in a blockchain network to access the data needed to verify transactions and blocks. This characteristic is crucial for maintaining the network's decentralization and trustlessness, enabling nodes to independently verify the blockchain's history and state. DAS is a method for verifying data availability without downloading the complete dataset, and is particularly suitable for lightweight nodes with limited storage capacity.

The concept of DAS was first introduced by Al-Bassam et al. [22]. In a DAS scheme, a potentially malicious block proposer encodes the block's contents into a short commitment *com* and a larger codeword π using an erasure code. The commitment *com* is placed in the block header. To verify that the block data is available, lightweight nodes randomly sample a few positions in π. If those samples can be obtained successfully, then with high probability the entire block data is available. Early implementations of this idea discussed DAS only informally [30–32], without precise security definitions or formal proofs. Hall-Andersen et al. [23] addressed this gap by formally defining DAS as a cryptographic primitive and introducing erasure-code commitments as a way to guarantee data availability. They proved that a secure DAS scheme can be built from any erasure-code commitment that meets certain binding and availability properties.

2.2 Existing DAS Protocols

To improve the efficiency of DAS in practice, Hall-Andersen et al. developed the FRIDA [26], which applies the FRI protocol to data availability. FRIDA avoids a trusted setup and achieves communication complexity that grows polylogarithmically with the block size. It was among the first schemes to explicitly integrate an IOPP into a DAS design, demonstrating that the FRI protocol can be adapted for checking data availability.

In parallel, Wagner et al. [33] presented PeerDAS, a scheme intended for integration into Ethereum. PeerDAS uses RS codes in combination with KZG polynomial commitments to enable sampling-based data availability checks for light clients. Their work describes optimizations for selecting random sample columns and verifying the corresponding commitments efficiently.

Despite these advances, existing DAS protocols still face scalability challenges. Even schemes with polylogarithmic complexity (such as FRIDA and PeerDAS) incur substantial communication and verification costs when block sizes become very large. Moreover, static choices of code rate or sampling density may be suboptimal under changing network conditions or adaptive adversaries. These limitations motivate ISTIRDA. ISTIRDA introduces adaptive folding and dynamic rate adjustment to tune the coding and sampling process for the data scale and threat model. In effect, ISTIRDA retains the security guarantees of IOPP-based DAS while significantly lowering communication and verification overhead in high-throughput, large-block settings. To highlight the advantages of our proposed scheme, Table 1 presents a comparison of existing schemes.

Table 1: Structured comparison of DAS schemes

Scheme	Trusted setup	Code family	Rate adaptivity	Comm.	Verifier time
Naive	No	RS/none	None	$\Omega(D)$	High
Merkle	No	Hash	None	$O(\lambda \log D)$	Low
PeerDAS	Yes	RS + KZG	Limited	polylog(D)	Low
FRIDA	No	RS/FRI	Fixed	polylog(D)	Good
ISTIRDA	No	RS/FRI-style	**Dynamic**	polylog(D) (smaller const.)	Best for $D \geq 16$ MB

3 Preliminaries

3.1 Interactive Oracle Proofs of Proximity

Key parameters. Let $L \subseteq \Sigma^n$ be a language over alphabet Σ. For $x, y \in \Sigma^n$, the Hamming distance is $d(x, y) = \left|\{i \in [n]: x_i \neq y_i\}\right|$. Write $d(x, L) = \min_{y \in L} d(x, y)$ and define the proximity set $L_\varepsilon = \{x \in \Sigma^n: d(x, L) \leq \varepsilon n\}$.

Completeness. If $x \in L$, an honest P makes V accept with probability at least $1 - \delta$.

Soundness. If $x \notin L_\varepsilon$, any P' convinces V with probability at most γ.

Query complexity. $Q = \sum_{i=1}^{r} q_i \ll n$.

Fig. 1 depicts the workflow of an IOPP. IOPP is an interactive protocol between a prover P and a verifier V that certifies $x \in L_\varepsilon$ with sublinear query access to x. The protocol runs for r rounds with per-round query budgets $q_1, \ldots, q_r$. In round i, $c_i = V_i(m_1, \ldots, m_{i-1})$, $m_i = P_i(c_i, x)$, $Q_i \subseteq [n]$, $|Q_i| \le q_i$, and V reads $\{x_j\}_{j \in Q_i}$ and checks consistency with m_i. After r rounds V accepts if all checks pass.

In this paper, the sequence $\{r_i\}$ jointly drives adaptive folding and dynamic code-rate adjustment, contracting the evaluation domain over rounds and reducing communication and verification cost so that data availability can be certified with sublinear read access.

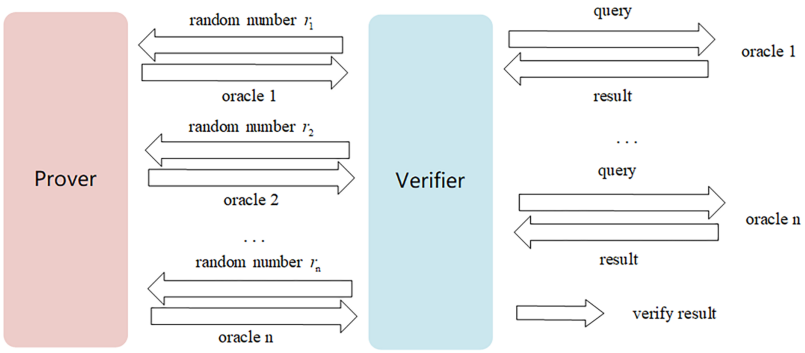

Figure 1: Workflow of an interactive oracle proofs of proximity

3.2 Reed–Solomon Codes

As visualized in Fig. 2, a message polynomial of degree strictly less than k is evaluated at n pairwise distinct field points to produce n coded symbols; because RS codes are maximum distance separable, any k symbols suffice to reconstruct the message.

Let $\mathbb{F}_q$ be a finite field and let $\alpha_1, \ldots, \alpha_n \in \mathbb{F}_q$ be pairwise distinct. An (n, k, d) RS code encodes a message polynomial $m(x) \in \mathbb{F}_q[x]$ with $\deg m < k$ (equivalently, $\deg m \le k - 1$) by evaluation: $c = \big(m(\alpha_1), \ldots, m(\alpha_n)\big)$, $R = \frac{k}{n}$. Since RS codes are maximum distance separable, the minimum (Hamming) distance is $d_{\min} = n - k + 1$ and the unique-decoding radius is $t = \lfloor (n-k)/2 \rfloor$. Equivalently, encoding admits the cyclic form $c(x) = m(x)\, x^{n-k} mod h(x)$, for a suitable generator polynomial $h(x)$, where $n - k$ is the number of parity symbols.

For illustration, take $n = 8$ and $k = 4$. Then $R = \frac{1}{2}$ and $d_{\min} = n - k + 1 = 5$, so up to $t = \lfloor (d_{\min} - 1)/2 \rfloor = \lfloor (n-k)/2 \rfloor = 2$ symbol errors can be uniquely corrected. Increasing redundancy (i.e., decreasing k/n) improves error-correction capability but raises communication and storage costs. In this paper, dynamic code-rate adjustment adapts k/n to balance target robustness against resource budgets, selecting appropriate rates across rounds and data scales.

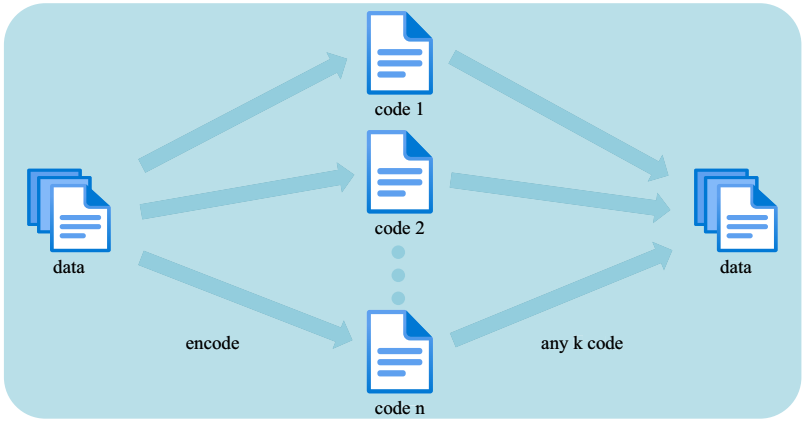

Figure 2: Illustration of RS codes

4 Our Proposed DAS Protocol

4.1 Overview

Fig. 3 illustrates the end-to-end flow of the proposed protocol: The original data is first Reed-Solomon encoded to obtain the codeword π, which is then bound to the block header commitment com. This interaction phase then proceeds, progressing by round i. The prover publicly publishes a queryable view π_i (derived from π through folding/state transfer). The verifier, comprised of two submodules, $V_1 \rightarrow \text{tran} \rightarrow V_2$, collaborates to initiate sparse queries on π_i and perform algebraic and cross-round consistency checks under the constraints of the public commitment com. tran is responsible for transferring/contracting the evaluation domain between rounds. Once all checks pass, the extractor extra combines the commitment with the sampled symbols to recover the original data. If any step fails, the data is deemed unusable.

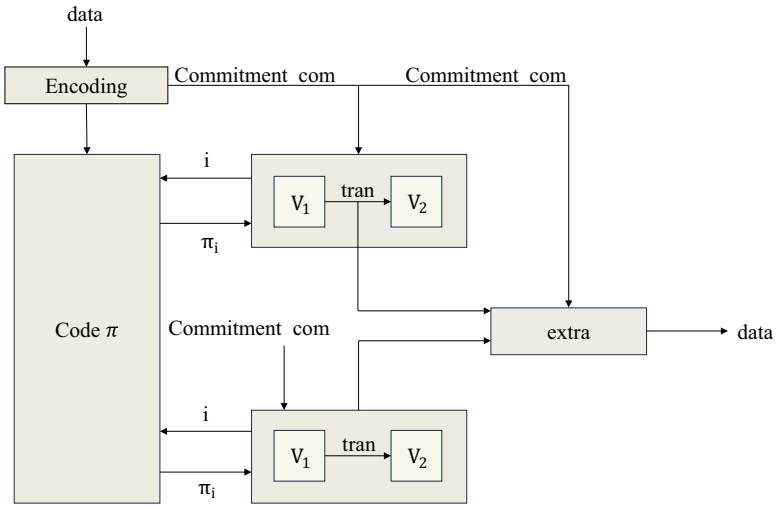

Figure 3: Flow of the data availability scheme

In what follows, we refine STIR [29] by removing redundant steps and introducing two mechanisms: adaptive folding and dynamic code-rate adjustment. Unlike STIR's fixed folding schedule, ISTIR selects the folding ratio per round based on the remaining evaluation domain, residual polynomial degree,

and security/redundancy targets, thereby avoiding late-round over-redundancy and premature domain contraction; this reduces both evaluation points and communication, especially at large data scales. The dynamic code-rate rule re-optimizes the rate at each recursion rather than letting it decay on a fixed schedule, balancing early-round bandwidth savings with stronger late-round soundness. We then analyze the resulting complexity parameters, which serve as performance indicators for ISTIRDA.

We further prove that ISTIR satisfies opening consistency under our model, ensuring transcripts remain well-structured across rounds, either close to a valid codeword or rejected, thus precluding attacks that exploit recursive folding. This security foundation enables the standard transformation from ISTIR to an erasure-code commitment and, following FRIDA's framework [26], to a complete DAS scheme, which we call ISTIRDA.

4.2 Construction of Our Protocol

Table 2 lists the protocol parameters and definitions. The specific protocol execution steps are as follows.

Table 2: Protocol parameters and definitions

Symbol	Description
F	A finite field.
M	The number of iterations ($M \in \mathbb{N}$).
d	Initial degree parameter ($d = 2^x$ for some $x \in \mathbb{N}$).
$k_0, \ldots, k_M$	Folding parameters ($k_i \in \mathbb{N}$, each a power of 2), satisfying $d \geq \prod_i k_i$.
$L_0, \ldots, L_M$	Evaluation domains ($L_i \subseteq F$), each a multiplicative subgroup of F^* with $\lvert L_i \rvert > d / \prod_{j \leq i} k_j$.
$t_0, \ldots, t_M$	Parameters for repeated sampling ($t_i \in \mathbb{N}$).
u	Additional repetition parameter for outer-domain sampling ($u \in \mathbb{N}$).
d_i	Auxiliary variable defined as $d_i := \prod_{j<i} k_j$ for each $i \in \{0, \ldots, M\}$.

Initialization. Define function $f_0 : L_0 \to F$ as a queried oracle. For an honest execution, the condition $f_0 \in RS[F, L_0, d_0]$ holds true. Thus, the prover can respond accurately to oracle queries about polynomial f_0. This polynomial belongs to the space $F^{<d_0}[X]$ and is restricted to domain L_0.

Initial Folding Step. The verifier randomly selects a folding scalar r_0^{fold} from the field F and transmits it.

Interactive Protocol Rounds. For round index $i = 1, \ldots, M$:

(a) Prover Polynomial Folding Transmission: The prover computes and submits the folded function $h_i : L_i \to F$. Under honesty, h_i corresponds exactly to the evaluated folding polynomial $\hat{h}_i := \text{PolyFold}(f_{i-1}, k_i, r_{i-1}^{\text{fold}})$ on domain L_i.

(b) The verifier selects random points: $r_{i,1}^{\text{out}}, \ldots, r_{i,u}^{\text{out}}$ from domain L_i outside previously queried positions.

(c) For these out-of-domain queries: The verifier receives responses $\beta_{i,1}, \ldots, \beta_{i,u}$ from the prover. Under honest conditions, each response is computed as $\beta_{i,j} = g_i(r_{i,j}^{\text{out}})$.

(d) ISTIR-specific Communication: The verifier sends a random scalar $r_i^{\text{ISTIR,comm}} \in F$ and points for queries $r_{i,1}^{\text{ISTIR,query}}, \ldots, r_{i,t_i}^{\text{ISTIR,query}} \in L_{i-1}$.

(e) The prover transmits a prediction message, denoted $\text{Fill}_i : \{0,1\}^* \to F$, which is defined as a polynomial. In the honest execution, this message is constructed by the prover as $H_i = \{r_i^{\text{ISTIR,comm}}, r_{i,1}^{\text{ISTIR,query}}, \ldots, r_{i,t_i}^{\text{ISTIR,query}}\}$, and $h_i' = \text{PolyQuotient}(h_i, H_i)$, and satisfies $\text{Fill}_i(r_{i,j}^{\text{ISTIR,query}}) = h_i'(r_{i,j}^{\text{ISTIR,query}})(\text{if } r_{i,j}^{\text{ISTIR,query}} \in L_i)$.
Additionally, in an honest execution, the prover computes and transmits the polynomial

$f_i' \in F^{<d_i}[X]$ whose degree is corrected with respect to the challenge and folding set: $f_i' = \mathrm{DegCor}\big(d_i, r_i^{\mathrm{ISTIR,comm}}, h_i, d_i - |H_i|\big)$. With the current round complete, the protocol transitions into the next phase at index $i+1$.

Final Round. At this stage, the prover outputs a polynomial $\rho \in F^{<d_M}[X]$ consisting of d_M coefficients. When acting honestly, the polynomial satisfies $\hat{\rho} = \mathrm{Fold}\big(f_M, k_M, r_M^{\mathrm{fold}}\big)$.

Verifier Decision Phase. The verifier executes the following verification steps:

(a) Iterative Verification Procedure: For $i = 1, \ldots, M$:

i. For each $j \in [t_{i-1}]$, the verifier evaluates $\mathrm{Fold}\big(f_{i-1}, k_{i-1}, r_{i-1}^{\mathrm{fold}}\big)\big(r_{i-1,j}^{\mathrm{ISTIR,query}}\big)$. To do so, f_{i-1} must be accessed at all k_{i-1} evaluation points in L_{i-1}, where the relation $x^{k_i-1} = r_{i-1,j}^{\mathrm{ISTIR,query}}$ holds.

ii. Construct the query set $H_i = \{r_i^{\mathrm{ISTIR,comm}}, r_{i,1}^{\mathrm{ISTIR,query}}, \ldots, r_{i,t_i-1}^{\mathrm{ISTIR,query}}\}$, and define a response mapping $\mathrm{Ans}_i : H_i \to F$ such that $\mathrm{Ans}_i(r_{i,j}^{\mathrm{ISTIR,comm}}) = \beta_i$, and for each j, we have $\mathrm{Ans}_i(r_{i,j}^{\mathrm{ISTIR,query}}) = \mathrm{Fold}(f_{i-1}, k_{i-1}, r_{i-1}^{\mathrm{fold}})(r_{i-1,j}^{\mathrm{ISTIR,query}})$. Based on these assignments, the verifier virtually computes the polynomial $h_i' = \mathrm{Quotient}(h_i, H_i, \mathrm{Ans}_i)$.

iii. Define a virtual oracle function f_i': $f_i' = \mathrm{DegCor}\big(d_i, r_i^{\mathrm{ISTIR,comm}}, h_i, d_i - |H_i|\big)$.
In practice, any query to f_i' is redirected to h_i if the input point is not in H_i, or to Fill_i otherwise.

(b) Consistency Check for the Final Folding Step:

i. Randomly select evaluation points $r_1^{\mathrm{final}}, \ldots, r_m^{\mathrm{final}} \in L_M$.

ii. For each $j \in [m]$, verify that $\rho\left(r_j^{\mathrm{final}}\right) = \mathrm{Fold}\left(f_M, k_M, r_M^{\mathrm{fold}}\right)\left(r_j^{\mathrm{final}}\right)$ holds.

iii. Cross-validate with Ans_i: for all $i \in \{1, \ldots, M\}$ and $x \in H_i \cap L_i$, evaluate $h_i(x)$ and ensure $h_i(x) = \mathrm{Ans}_i(x)$.

Algorithm 1 gives the Fiat-Shamir compact form of the interactive protocol described above, where the challenge is derived in the random oracle model via $r_t = \mathrm{Hash}(\mathsf{Tr})$. Throughout, we define $f_t \equiv k_t$ (folding factor/rate), $\tau_{\min}$ denotes the decoding margin, requiring $(1-\rho)/2 \geq \tau_{\min}$.

Algorithm 1: ISTIR/IOPP (FS-compact)

Require: data D, field $\mathbb{F}_q$, code (n_0, k_0) with $\rho_0 = k_0/n_0$, domain L_0, terminal size $N_{\min}$, margin $\tau_{\min}$
Ensure: Accept or Reject

Pack $D \to m$; $\pi_0 \leftarrow$ RS_Encode(m); $com \leftarrow$ Commit(π_0); $\mathsf{Tr} \leftarrow (com)$; $t \leftarrow 0$
while $|L_t| > N_{\min}$ **do**
  $r_t \leftarrow \mathrm{Hash}(\mathsf{Tr})$ ▷ FS challenge
  Choose (f_t, ρ_{t+1}, q_t) with $|L_t|/f_t \geq N_{\min}$ and $(1-\rho_{t+1})/2 \geq \tau_{\min}$
  $(\pi_{t+1}, \mathcal{W}_t) \leftarrow Fold(\pi_t, f_t, r_t)$; $Tr \leftarrow Tr \,\|\, (\pi_{t+1}, \mathcal{W}_t)$
  Sample $S_t \subseteq L_t$, $|S_t| = q_t$ via r_t; open $\pi_t|_{S_t}, \mathcal{W}_t|_{S_t}$ vs. com; **if** fail **then return** Reject
  $L_{t+1} \leftarrow L_t/f_t$; $t \leftarrow t+1$
end while
Final low-degree/proximity check on π_t over L_t; **if** fail **then** Reject
return Accept

Complexity Parameters.

The protocol involves a total of $2M+1$ communication rounds. During each round indexed by $i \in \{1, \ldots, M\}$, the prover transmits the folded polynomial h_i with length $|L_i|$, a set of out-of-domain responses $\beta_{i,1}, \ldots, \beta_{i,u}$, and the oracle function Fill_i, which contains at most $t_{i-1} + u$ symbols.

In the final stage, the prover additionally sends $d_M = d/\prod_{j=0}^{M} k_j$ field elements. Hence, the total proof size accumulates to $\sum_{i=1}^{M} (|L_i| + u + t_{i-1}) + \frac{d}{\prod_{j=0}^{M} k_j}$. On the verifier's side, it processes t_0 queries, each requiring the evaluation of k_0 points. Since the same set of k_0 points is always accessed as a unit, this allows for treating them as a single composite symbol, and the input query cost in this alphabet is thus t_0. For subsequent rounds $i \in \{1, \ldots, M\}$, the verifier issues t_i queries to the folded polynomial: $\mathrm{Fold}(f_{i-1}, k_{i-1}, r_{i-1}^{\mathrm{fold}})$, which necessitates reading k_{i-1} grouped symbols from the previous function f_{i-1}.

In addition, the verifier may submit up to $|H_i| \leq t_{i-1}$ individual queries to f_i. When $i = 1$, these queries directly access terminal data. For $i > 1$, however, f_i represents a virtual layer, where each access is rerouted either to a single point evaluation of Fill_{i-1}, or to a corresponding position in h_{i-1}, which again involves accessing k_{i-1} symbols. Because the k_i symbols are accessed simultaneously, they can be compressed into a single higher-level symbol. Therefore, the query complexity of the proof string is $2 \cdot \sum_{i=1}^{M} t_i$.

4.3 Opening Consistency

4.3.1 Defining the Suitable Transcript Set

In an interactive ISTIR protocol, a partial transcript is any prefix of the interaction between the prover and the verifier.

The lucky set consists of partial prover transcripts that satisfy algebraic collision and proximity conditions. In round i the prover sends H_{i-1}; let H_i^* be a nearest codeword. If there exists an algebraic hash $\mathfrak{h}_{\rho_i}$ such that $\mathfrak{h}_{\rho_i}(H_{i-1})$ collides with $\mathfrak{h}_{\rho_i}(H_i^*)$, and the designated blockwise distance and post-folding proximity conditions hold, then the transcript up to round i is lucky.

Formally, the lucky set contains all partial transcripts for which either $\mathfrak{h}_{\rho}(H_{i-1}) = \mathfrak{h}_{\rho}(H_{i-1}^*)$ or the folding step honestly reduces the distance.

Definition 1 (Lucky Set for ISTIR). Define the unique decoding radius as $\delta^* = (1 - \rho)/2$. The collection of partial prover transcripts referred to as the lucky set is given by Lucky = LuckyColl ∪ LuckyDist, with the components defined by the following steps:

A partial transcript of the form $(H_0, \rho_1, \ldots, H_{i-1}, \rho_i)$ is said to belong to LuckyColl if the following two criteria are satisfied simultaneously:

(a) The symbol vector H_{i-1} resides within the decoding radius of the ball around code C_{i-1}, i.e., $\delta^B(H_{i-1}, C_{i-1}) \leq \delta^*$, where $H_{i-1}^* \in C_{i-1}$ denotes the nearest codeword.

(b) There exists some element $u_i \in L_i$ such that the evaluations of the hash function coincide, i.e., $\mathfrak{h}_{\rho_i}[H_{i-1}](u_i) = \mathfrak{h}_{\rho_i}[H_{i-1}^*](u_i)$, while for some $u_{i-1} \in L_{i-1}$, the deviation is nonzero: $q(u_{i-1}) = u_{i-1} \cdot (H_{i-1}(u_{i-1}) - H_{i-1}^*(u_{i-1})) \neq 0$.

Furthermore, a partial transcript $(H_0, \rho_1, \ldots, H_{i-1}, \rho_i, C_i)$ is classified into LuckyDist if at least one of the following holds: either the previous layer lies exactly on the boundary, i.e., $\delta^B(H_{i-1}, C_{i-1}) = \delta^*$, and the hash proximity satisfies $\delta(\mathfrak{h}_{\rho_i}[H_{i-1}], C_i) \leq \delta^*$; or alternatively, the hashed distance is strictly closer, i.e., $\delta(\mathfrak{h}_{\rho_i}[H_{i-1}], C_i) < \delta^B(H_{i-1}, C_{i-1}) \leq \delta^*$.

The definition of the bad set is relative to the lucky set. A partial transcript is considered part of the bad set if it is not contained in the lucky set and satisfies certain specific "bad" conditions. These conditions include: the prover's final oracle message does not match the intended codeword; or in some round, the oracle deviates from its expected codeword; or the oracle is closer to an incorrect codeword and fails to preserve the folding structure.

Definition 2 (Bad Set in ISTIR). The unique decoding radius is given by $\delta^* = (1-\rho)/2$. We define the Bad set of a partial transcription $(H_0, \rho_1, H_1, \ldots, \rho_r, H_r)$ as the set of all transcripts for which all prover round prefixes are in the Lucky set and which also meet at least one of the following criteria:

(a) $H_r \neq C_r$.
(b) there exists some $i \in \{0, \ldots, r-1\}$ such that $\delta^B(H_i, C_i) > \delta^*$.
(c) for every $i \in \{0, \ldots, r-1\}$, the inequality $\delta^B(H_i, C_i) \leq \delta^*$ holds; however, there exists some $i \in [r]$ such that $H_i^* \neq \mathfrak{h}_{\rho_i}[H_{i-1}^*]$, where H_i^* denotes the unique nearest codeword of H_i.

4.3.2 Four Fundamental Properties of Opening Consistency

No Luck. We need to show that for every $i \in [k]$ (where k denotes the total number of rounds in ISTIR), and for any subset of $2i-1$ elements of partial verifier transcripts T, the probability that their extension reaches the lucky set is bounded.

During every round of interaction, the verifier selects a random challenge, and the prover replies based on that challenge. By analyzing the folding of functions, hash operations, and codeword distances within the ISTIR protocol, we can compute the probability that a transcript is extended into the lucky set under certain conditions. For example, given $T = (H_0, \rho_1, \cdots, H_{i-1})$, when the verifier issues a new challenge ρ_i, we analyze the conditions under which $T \circ \rho_i$ enters the lucky set. This may involve the behavior of function H_{i-1} and its nearest codeword under the hash operation, as well as considerations of blockwise and other relevant distance conditions. Ultimately, we derive an upper bound for $\Pr_{\rho_i}[T \circ \rho_i \in Lucky]$.

Lemma 1 (No Luck). ISTIR satisfies the No Luck property, and $\varepsilon_1 \leq \frac{2(F-1)|L_0|}{|\mathbb{F}|}$.

Proof. Fix $i \in [r]$, and let $T = (H_0, \rho_1, \rho_2 \ldots, H_{i-1})$ be a partial verifier transcript. Consider $\rho_i \in \mathbb{F}$ sampled uniformly at random. We aim to upper-bound the probability that $T \circ \rho_i \in Lucky$. For the *LuckyColl* case, we claim: $\Pr_{\rho_i}[T \circ \rho_i \in LuckyColl] \leq \dfrac{(F-1)|L_0|}{|\mathbb{F}|}$. To prove this, we bound the probability over each input $u_i \in L_i$ causing $LuckyColl_{u_i}$. By union bound: $\Pr_{\rho_i}[T \circ \rho_i \in LuckyColl] \leq \sum_{u_i \in L_i} \Pr_{\rho_i}[LuckyColl_{u_i}]$.

In what follows, fix an arbitrary $u_i \in L_i$ and $\Pr_{\rho_i}[LuckyColl_{u_i}] \leq \dfrac{F-1}{|\mathbb{F}|}$. Since $|L_i| \leq |L_0|$, this suffices. To define $LuckyColl_{u_i}$, as the event that arises precisely when the following two conditions are met: $\mathfrak{h}_{\rho_i}[H_{i-1}](u_i) = \mathfrak{h}_{\rho_i}[H_{i-1}^*](u_i)$ and $\{(u_{i-1}, H_{i-1}(u_{i-1}))\}_{u_{i-1} \in q^{-1}(u_i)} \neq \{(u_{i-1}, H_{i-1}^*(u_{i-1}))\}_{u_{i-1} \in q^{-1}(u_i)}$. This implies two distinct polynomials of degree at most $(F-1)$ agree on a point. Let $p, p^* \in \mathbb{F}^F$ denote the coefficient vectors of H_{i-1} and H_{i-1}^* under the hash operation $\mathfrak{h}_{\rho_i}$. If we can show $p \neq p^*$, the proof is complete. Let $u_{i-1,1}, \ldots, u_{i-1,F} \in L_{i-1}$ be the preimage set of u_i. Define the associated Vandermonde matrix $\mathbf{V}_{u_i} \in \mathbb{F}^{F \times F}$ with j-th row $(1, u_{i-1,j}, \ldots, u_{i-1,j}^{F-1})$, which is invertible.

Define $h_{u_i}, h_{u_i}^* \in \mathbb{F}^F$ as the evaluation vectors of H_{i-1} and H_{i-1}^* on $u_{i-1,1}, , \ldots, u_{i-1,F}$, respectively. If $LuckyColl_{u_i}$ occurs, then $h_{u_i} \neq h_{u_i}^*$, and thus: $p = \mathbf{V}_{u_i}^{-1} h_{u_i} \neq \mathbf{V}_{u_i}^{-1} h_{u_i}^* = p^*$, so $p \neq p^*$, which contradicts their equality under $\mathfrak{h}_{\rho_i}$. Therefore, the probability that this happens for any u_i is at most $1/|\mathbb{F}|$ per degree-F collision, and across $(F-1)$ degrees we get the desired bound. This completes the proof. □

Bad is Rejected. Suppose $T = (H_0, \rho_1, \cdots, H_k)$ be a transcript with $T \in Bad$. Our goal is to show that the probability $\Pr_{\rho_{k+1}}[\mathbb{V}_{H_0, H_1, \cdots, H_k}(\rho_1, \cdots, H_k, \rho_{k+1}) = 1]$ is bounded.

If a transcript T lies in the bad set, then by the definition of *Bad*, it either contains a final prover message that is inconsistent with the code, or it contains some intermediate round where distance-related conditions or folding operations do not permit switching arguments. We analyze the verifier's acceptance probability across these cases. For example, if the final prover message does not lie within the code, then according to

the ISTIR verification rule, the verifier's probability of accepting during the final check should be negligibly small. More generally, for the other types of badness, we can analyze the verifier's decision process and the structure of the code to upper-bound this acceptance probability.

Lemma 2 (Bad is Rejected). ISTIR satisfies the property that any transcript labeled as "Bad" will be rejected by the verifier during the opening consistency process.

Proof. Consider $T = (H_0, \rho_1, H_1, \ldots, \rho_r, H_r)$ such that $T \in Bad$, as defined in Definition 2. Let $\rho_{r+1} = u_0 \xleftarrow{\$} \mathcal{L}_0$ be a freshly sampled point from the domain $\mathcal{L}_0$, and execute the verifier's procedure on the extended transcript $T \circ u_0$. The goal is to establish an upper bound on the probability that the verifier accepts. We proceed by analyzing two scenarios. Firstly, suppose $H_r \notin \mathcal{C}_{r_\circ}$. In this situation, the verifier clearly rejects.

Now consider the case where $H_r \in \mathcal{C}_{r_\circ}$. According to the definition of *Bad*, there must exist some index $i \in \{0, \ldots, r-1\}$ such that the decoding distance exceeds the unique radius, i.e., $\delta^B(H_i, C_i) > \delta^*$, or alternatively, all such distances satisfy $\delta^B(H_i, C_i) \leq \delta^*$ for $i \in \{0, \ldots, r-1\}$, but there is some $i \in [r]$ for which the decoding of H_i disagrees with the intended image under the hash, i.e., $H_i^* \neq \mathfrak{h}_{\rho_i}(H_{i-1}^*)$, where H_i^* is the unique codeword closest to H_i. This completes the proof. □

Suitability is Close. Let $T = (H_0, \rho_1, \ldots, H_k)$ denote a transcript that satisfies the conditions of both *Bad* and *Lucky* sets. We aim to show that H_0 (or more generally, the function or codeword involved) is within decoding radius, i.e., $\delta(C, H_0) \leq \delta^*$.

According to the definitions of suitable transcripts, as well as those of *Lucky* and *Bad* sets, we analyze how the ISTIR protocol operations in each round affect the distance between functions and codewords. We demonstrate that all the operations composing a suitable transcript induce only limited growth in codeword distance. For example, in a particular round, the folding or hashing operations may only slightly alter the proximity between functions and their corresponding codewords. By combining such structural and distance-preserving properties, and leveraging the metric's definition, we conclude that the transcript satisfies the "Suitability is Close" property.

Lemma 3 (Suitable is Close). ISTIR satisfies the "suitability is close" of opening consistency.

Proof. Suppose $T = (H_0, \rho_1, \ldots, H_r)$ is suitable to both the bad set and the lucky set. This suitability implies that $T \notin Bad$, and none of its prefixes belong to *Lucky*. According to the definitions of *Bad* and *Lucky*, we can conclude that $\delta^B(H_i, C_i) \leq \delta^*$ holds. Then, by applying Lemma 2 from [26], it follows that $\delta(H_i, C_i) \leq \delta^*$. In particular, this inequality also holds for H_0, which completes the proof. □

Inconsisteny is Rejected. Suppose $T = (H_0, \rho_1, \ldots, H_k)$ is suitable to both the bad set and the lucky set. Suppose $H_0^* \in C$ is the only codeword nearest to H_0. If there exists an index $j \in Q_0(T \circ \rho_{k+1})$ such that $H_{0j} \neq H_{0j}^*$,then it must be shown that the transcript does not satisfy the verifier's checks: $V^{H_0, H_1, \ldots, H_k}(\rho_1, \ldots, H_k, \rho_{k+1}) = 0$. When an inconsistency arises with respect to the unique closest codeword, we analyze the verifier's decision process and the verification rules of the ISTIR protocol. Since the verifier performs function queries and corresponding checks in each round, we demonstrate—by analyzing these operations and studying the inconsistency detection mechanism—that the verifier is capable of identifying such inconsistencies and rejecting the transcript.

Lemma 4 (Inconsistency is Rejected). ISTIR satisfies the property of inconsistency rejection for opening consistency.

Proof. Recall that suitability implies $T \notin Bad$ and none of its prefixes lie in $Lucky$. Suppose $H_i^* \in C_i$ is the only codeword nearest to H_i. By the definition of Bad, we know $H_i^* = \mathfrak{h}_{\rho_i}[H_{i-1}^*]$ for every $i \in [r]$. Now, consider using $\rho_{r+1} = u_0 \in \mathcal{L}_0$ to complete the transcript T. For each $i \in [r]$, let u_i be the value queried by the ISTIR verifier, defined by $u_i = q(u_{i-1})$.

Now, there must exist a query location $x \in Q_0(T \circ u_0) \subseteq \mathcal{L}_0$ such that $H_0^*(x) \neq H_0(x)$, and we aim to prove that the completed transcript $T \circ u_0$ is rejected by the verifier. Let i_0^* be the smallest index in $\{0, \ldots, r\}$ such that for a queried location $u_i' \in \mathcal{L}_i$, the verifier's query yields $H_{i_0}(u_{i_0}') \neq H_{i_0}^*(u_{i_0}')$. Notice this index exists since we assumed $H_r = H_r^*$.

Moreover, since there exists a query $x \in Q_0(T \circ u_0) \subseteq \mathcal{L}_0$ with $H_0^*(x) \neq H_0(x)$, we have $i_0 > 0$. Such an assumption cannot be reconciled with the fact that $T \circ u_0$ is accepted. Then, we get the equality $\forall i \in [r], H_i(u_i) = \text{Interpolate}\left(\rho_i, \{(u_{i-1}, H_{i-1}(u_{i-1})) \mid u_{i-1} \in q^{-1}(u)\}\right)$, where $Interpolate(z, U)$ receives $z \in \mathbb{F}$ and $U \subseteq \mathbb{F}^2$, putting the unique polynomial $P(z)$ of a degree less than $|U|$ that satisfies $P(x) = y$ for all $(x, y) \in U$. Therefore, we derive $\mathfrak{h}_{\rho_{i_0}}[H_{i_0-1}](u_{i_0}) = H_{i_0}(u_{i_0}) = H_{i_0}^*(u_{i_0})$. Furthermore, we have $H_{i_0}^*(u_{i_0}) = \mathfrak{h}_{\rho_{i_0}}[H_{i_0-1}^*(u_{i_0})]$, because for all i we have $H_{i_0}^* = \mathfrak{h}_{\rho_{i_0}}[H_{i_0-1}^*]$.Therefore $\mathfrak{h}_{\rho_{i_0}}[H_{i_0-1}](u_{i_0}) = \mathfrak{h}_{\rho_{i_0}}[H_{i_0-1}^*](u_{i_0})$.

By the definition of the minimal index i_0, there must also exist a query location $u_{i_0-1}' \in \mathcal{L}_{i_0-1}$ satisfying $q(u_{i_0-1}') = u_{i_0}$ and $H_{i_0-1}(u_{i_0-1}') \neq H_{i_0-1}^*(u_{i_0-1}')$. Therefore, we conclude $(H_0, \rho_1, \ldots, H_{i_0-1}, \rho_{i_0}) \in LuckyColl \subseteq Lucky$, which contradicts the suitability assumption of transcript T. This completes the proof. □

We formally state the opening consistency property of ISTIR in the next theorem.

Theorem 1 (Opening Consistency of ISTIR). ISTIR satisfies opening consistency relative to the formal definitions of *Bad* and *Lucky* provided in Definitions 1 and 2, with error bounds ε_1 and ε_2, where $\varepsilon_1 \leq \frac{2(F-1)|L_0|}{|F|}$ and $\varepsilon_2 \leq 1 - \delta^*$.

5 Performance

In this section, we conduct a detailed performance evaluation of ISTIRDA. Our evaluation metrics include communication cost, verification time and Time-to-Confidence (TTC). For communication cost and verification time, the schemes compared include Naive, Merkle, FRIDA, and ISTIRDA. Naive and Merkle are used as baseline schemes. Our experimental environment is built on a single machine, equipped with an Intel Core i7-9750H CPU running at 2.6 GHz, 16 GB of RAM, and a 64-bit Ubuntu 14.04 LTS operating system. We use Python 3.8 for implementation. This setup ensures reliable and accurate results. To illustrate system-level behavior, we measure TTC and compare only FRIDA and ISTIRDA. Because ISTIRDA targets permissionless public blockchains, we evaluate it in a controlled peer-to-peer emulation rather than a live network. One proposer process and 10–460 light-client processes run on a single physical host, and we inject latency and packet loss via Linux tc/netem to approximate wide-area blockchain conditions. This preserves the protocol's logical one-to-many dissemination while giving us strict control over network parameters. We report TTC, defined as the wall-clock time for a light client to reach the target availability confidence. TTC is measured under fixed protocol and network parameters (block size, sampling rate λ, quorum size k, round-trip latency) while varying only the number of light clients and the packet loss rate. A limitation is that single-host emulation is not a substitute for a geographically distributed deployment or a public testnet; Section 6 outlines ongoing work on multi-host and public Ethereum-compatible testnet evaluation.

5.1 Communication Cost

For different encoded data sizes $D = |\text{Data}|$, we compare the commitment size, encoding size, communication complexity per query, and the total communication complexity of being able to reconstruct the data with probability at least $1 - 2^{-40}$.

Fig. 4 presents a performance comparison among ISTIRDA, FRIDA, and other baseline schemes (Naive, Merkle) under different data sizes ($D = 1$, $D = 32$, $D = 128$ MB), with respect to four evaluation metrics: Commitment size (KB), Encoding (MB), Communication cost per query (KB), and Total communication

cost (MB). The communication cost incurred by ISTIRDA exhibits a significant advantage over that of the FRIDA scheme. Specifically, as the data size increases, the communication cost of FRIDA ranges from 1.25 to 2.46 times that of ISTIRDA. This advantage becomes even more pronounced in scenarios with large-scale data.

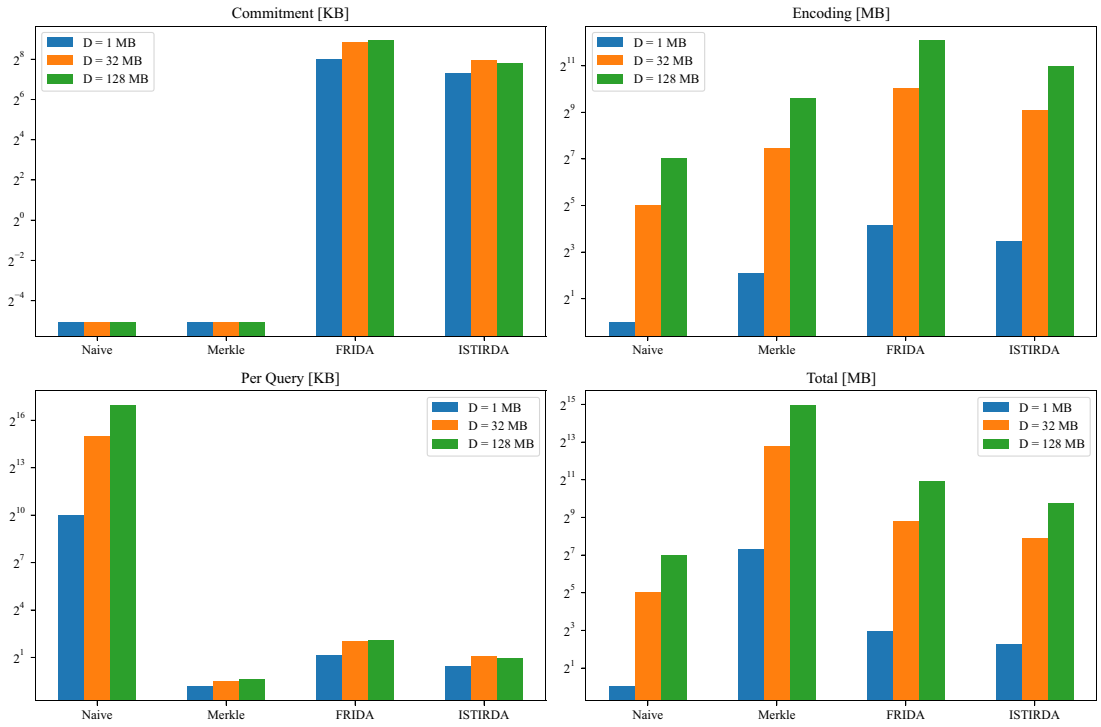

Figure 4: Performance comparison of data availability schemes across multiple metrics and data sizes

5.2 Time Cost

Prover computational time. In our tested dataset, the prover's computational time for ISTIRDA is slightly longer compared to FRIDA. According to measurements, its slowdown ranges from approximately 0.64 to 0.95 times. As shown in Table 3, when processing data of size |Data| = 16 MB, ISTIRDA takes 36.63 s to generate a proof sequentially, whereas FRIDA requires only 29.52 s.

Table 3: Prover and verifier computational costs for ISTIRDA and FRIDA under different data sizes

	1MB	16 MB	32 MB	64 MB	128 MB
Prover cost (s): ISTIRDA	3.83	36.63	63.01	98.69	210.31
Prover cost (s): FRIDA	3.97	29.52	59.37	147.21	320.89
Verifier cost (ms): ISTIRDA	2.67	3.90	3.93	3.98	4.15
Verifier cost (ms): FRIDA	3.18	3.82	4.37	4.77	5.06

Verifier computational time. The data shows that when the data size D = 1 MB, FRIDA's verification time is shorter than that of ISTIRDA. At D = 16 MB, the verification times of both schemes are comparable. However, for data sizes exceeding 16 MB, ISTIRDA's verification time becomes shorter than FRIDA's, and the gap continues to widen as the data size increases. The underlying reason for this phenomenon is that when

$D < 16$ MB, ISTIRDA's folding operation introduces computational overhead that outweighs its efficiency advantages. For larger data sizes, the folding operation's computational overhead increases slowly. Its growth rate is significantly lower than the verifier's data complexity growth. Therefore, for data sizes exceeding 16 MB, ISTIRDA's verification time is significantly lower than that of FRIDA.

5.3 Time-to-Confidence

Fig. 5 shows how the time to reach a confident decision varies with node count and packet loss. We observe that as node count and packet loss increase, FRIDA's TTC rises sharply, indicating that it becomes increasingly sensitive to load concentration. In contrast, ISTIRDA's TTC curve is much flatter, indicating that its distributed P2P collaborative architecture shares the load and maintains relatively fast convergence to confidence. Thus, under large-scale or degraded network conditions, ISTIRDA exhibits a clear robustness advantage over FRIDA.

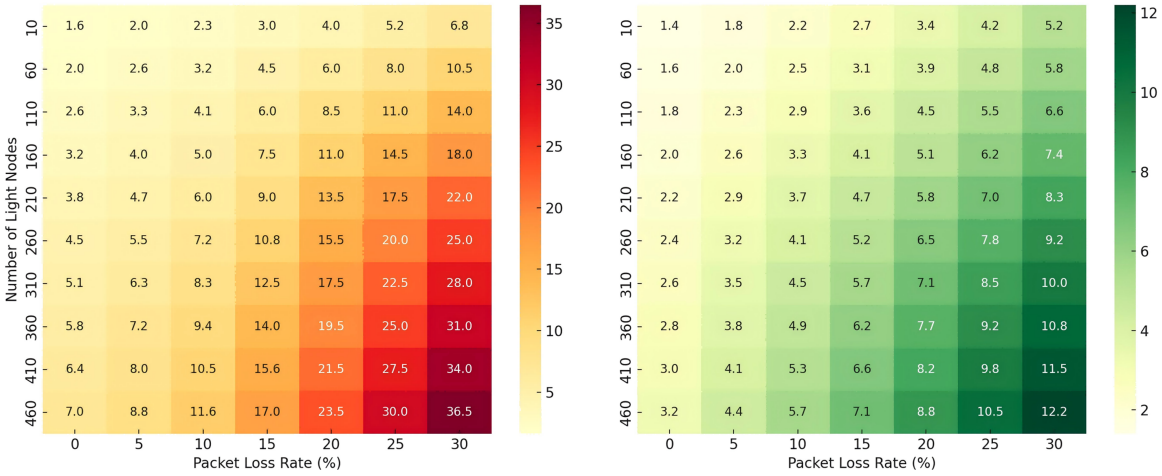

Figure 5: Heatmaps of TTC. The x-axis represents node count, the y-axis represents packet loss rate, and the color gradient indicates the time to reach a confident inference. (**a**) FRIDA; (**b**) ISTIRDA

6 Conclusion and Future Work

ISTIRDA improves upon DAS through rate-adaptive RS-IOPP (ISTIR), reducing the evaluation scope without sacrificing reliability. Experiments demonstrate that ISTIRDA significantly reduces communication and verifier time compared to Naive, Merkle, and FRIDA, especially for large data sizes (e.g., >16 MB), while only adding a small amount of prover overhead. TTC results demonstrate that, under matched reliability and typical bandwidth/loss conditions, ISTIRDA reaches the target availability confidence level faster, and its advantage increases with increasing block size or deteriorating network conditions. This makes ISTIRDA a reliable, scalable, and secure choice for lightweight clients.

This work demonstrates both the theoretical soundness and the practical feasibility of ISTIRDA for public blockchains using a controlled peer-to-peer emulation. Future efforts will move beyond single-machine `tc/netem` simulations toward physical multi-node testbeds and public testnets instrumented for heterogeneous latency, churn, and loss. A prototype light client integrated into an existing blockchain client (e.g., Geth or Nethermind) will enable end-to-end measurements of bandwidth, CPU/memory footprint, TTC, and failure modes under live consensus dynamics. The comparative scope will expand to emerging data availability designs, including 2D erasure-coded DA chains and ML-guided adaptive sampling to dynamically optimize parameters like folding ratio in response to network conditions, thereby maximizing

efficiency while preserving soundness. These evaluations will stress-test scalability and robustness across data scales and adversarial conditions. Finally, compatibility with Layer-2 systems will be explored by sampling published batch data and exposing a lightweight API for provers and light clients, with experiments mapping batch size, sampling rate, and security margins to L1/L2 throughput and latency.

Acknowledgement: The authors thank the Key Lab of Education Blockchain and Intelligent Technology, Ministry of Education for supporting this study.

Funding Statement: This work was supported in part by the Research Fund of Key Lab of Education Blockchain and Intelligent Technology, Ministry of Education (EBME25-F-08).

Author Contributions: Conceptualization, Jiaxi Wang; methodology, Jiaxi Wang; validation, Jiaxi Wang; investigation, Wenbo Sun; resources, Ziyuan Zhou; writing—original draft preparation, Jiaxi Wang; writing—review and editing, Shihua Wu; visualization, Jiaxi Wang; supervision, Shan Ji; project administration, Jiang Xu; funding acquisition, Shan Ji. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: Not applicable.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Chang V, Baudier P, Zhang H, Xu Q, Zhang J, Arami M. How Blockchain can impact financial services—the overview, challenges and recommendations from expert interviewees. Technol Forecast Soc Change. 2020;158(6):120166. doi:10.1016/j.techfore.2020.120166.
2. Patel R, Migliavacca M, Oriani ME. Blockchain in banking and finance: a bibliometric review. Res Int Bus Finance. 2022;62(4–5):101718. doi:10.1016/j.ribaf.2022.101718.
3. Kowalski M, Lee ZWY, Chan TKH. Blockchain technology and trust relationships in trade finance. Technol Forecast Soc Change. 2021;166:120641. doi:10.1016/j.techfore.2021.120641.
4. Han Y, Fang X. Systematic review of adopting blockchain in supply chain management: bibliometric analysis and theme discussion. Intl J Prod Res. 2024;62(3):991–1016. doi:10.1080/00207543.2023.2236241.
5. Surucu-Balci E, Iris Ç, Balci G. Digital information in maritime supply chains with blockchain and cloud platforms: supply chain capabilities, barriers, and research opportunities. Technol Forecast Soc Change. 2024;198(16):122978. doi:10.1016/j.techfore.2023.122978.
6. Ren Y, Leng Y, Qi J, Sharma PK, Wang J, Almakhadmeh Z, et al. Multiple cloud storage mechanism based on blockchain in smart homes. Fut Gener Comput Syst. 2021;115(3):304–13. doi:10.1016/j.future.2020.09.019.
7. Miao J, Wang Z, Wu Z, Ning X, Tiwari P. A blockchain-enabled privacy-preserving authentication management protocol for Internet of Medical Things. Exp Syst Appl. 2024;237(1):121329. doi:10.1016/j.eswa.2023.121329.
8. Samadhiya A, Kumar A, Arturo Garza-Reyes J, Luthra S, del Olmo García F. Unlock the potential: unveiling the untapped possibilities of blockchain technology in revolutionizing Internet of medical things-based environments through systematic review and future research propositions. Inf Sci. 2024;661(14):120140. doi:10.1016/j.ins.2024.120140.
9. Zhou X, Huang W, Liang W, Yan Z, Ma J, Pan Y, et al. Federated distillation and blockchain empowered secure knowledge sharing for Internet of medical Things. Inf Sci. 2024;662(4):120217. doi:10.1016/j.ins.2024.120217.
10. Yan Z, Zhao X, Liu YA, Luo XR. Blockchain-driven decentralized identity management: an interdisciplinary review and research agenda. Inf Manag. 2024;61(7):104026.
11. Al Sibahee MA, Abduljabbar ZA, Ngueilbaye A, Luo C, Li J, Huang Y, et al. Blockchain-based authentication schemes in smart environments: a systematic literature review. IEEE Internet Things J. 2024;11(21):34774–96. doi:10.1109/jiot.2024.3422678.

12. Shen H, Wang T, Chen J, Tao Y, Chen F. Blockchain-based batch authentication scheme for internet of vehicles. IEEE Trans Veh Technol. 2024;73(6):7866–79. doi:10.1109/tvt.2024.3355711.
13. Wang C, Wang C, Shen J, Vasilakos AV, Wang B, Wang W. Efficient batch verification and privacy-preserving data aggregation scheme in V2G networks. IEEE Trans Vehicular Technol. 2025;74(8):12029–12041. doi:10.1109/tvt.2025.3552494.
14. Sun L, Wang Y, Ren Y, Xia F. Path signature-based XAI-enabled network time series classification. Sci China Inform Sci. 2024;67(7):170305. doi:10.1007/s11432-023-3978-y.
15. Rebello GAF, Camilo GF, de Souza LAC, Potop-Butucaru M, de Amorim MD, Campista MEM, et al. A survey on blockchain scalability: from hardware to layer-two protocols. IEEE Commun Surv Tutor. 2024;26(4):2411–58.
16. Yaish A, Qin K, Zhou L, Zohar A, Gervais A. Speculative denial-of-service attacks in ethereum. In: 33rd USENIX Security Symposium (USENIX Security 24). Philadelphia, PA, USA: USENIX Association; 2024. p. 3531–48.
17. Ren Y, Lv Z, Xiong NN, Wang J. HCNCT: a cross-chain interaction scheme for the blockchain-based metaverse. ACM Trans Multimed Comput Commun Appl. 2024;20(7):1–23. doi:10.1145/3594542.
18. Sheng X, Wang C, Shen J, Sattamuthu H, Radhakrishnan N. Verifiable private data access control in consumer electronics for smart cities. IEEE Consumer Electron Magaz. 2025;14(6):100–6. doi:10.1109/mce.2024.3524750.
19. Chatzigiannis P, Baldimtsi F, Chalkias K. SoK: blockchain light clients. In: Eyal I, Garay J, editors. Financial cryptography and data security. Cham, Switzerland: Springer International Publishing; 2022. p. 615–41. doi:10.1007/978-3-031-18283-9_31.
20. Zong J, Wang C, Shen J, Su C, Wang W. ReLAC: revocable and lightweight access control with blockchain for smart consumer electronics. IEEE Trans Consumer Electron. 2024;70(1):3994–4004. doi:10.1109/tce.2023.3279652.
21. Zamyatin A, Avarikioti Z, Perez D, Knottenbelt WJ. TxChain: efficient cryptocurrency light clients via contingent transaction aggregation. In: Garcia-Alfaro J, Navarro-Arribas G, Herrera-Joancomarti J, editors. Data privacy management, cryptocurrencies and blockchain technology. Cham, Switzerland: Springer International Publishing; 2020. p. 269–86. doi:10.1007/978-3-030-66172-4_18.
22. Al-Bassam M, Sonnino A, Buterin V, Khoffi I. Fraud and data availability proofs: detecting invalid blocks in light clients. In: Borisov N, Diaz C, editors. Financial cryptography and data security. Berlin/Heidelberg, Germany: Springer; 2021. p. 279–98. doi:10.1007/978-3-662-64331-0_15.
23. Hall-Andersen M, Simkin M, Wagner B. Foundations of data availability sampling. IACR Commun Cryptol. 2023;1(4):79. doi:10.62056/a09qudhdj.
24. Reed IS, Solomon G. Polynomial codes over certain finite fields. J Soc Ind Appl Math. 1960;8(2):300–4. doi:10.1137/0108018.
25. Kate A, Zaverucha GM, Goldberg I. Constant-size commitments to polynomials and their applications. In: Abe M, editor. ASIACRYPT 2010: Advances in Cryptology. Vol. 6477. Berlin/Heidelberg, Germany: Springer; 2010. p. 177–94. doi: 10.1007/978-3-642-17373-8_11.
26. Hall-Andersen M, Simkin M, Wagner B. FRIDA: data availability sampling from FRI. In: Reyzin L, Stebila D, editors. Advances in Cryptology–CRYPTO 2024. Cham, Switzerland: Springer Nature; 2024. p. 289–324 doi: 10.1007/978-3-031-68391-6_9.
27. Ben-Sasson E, Bentov I, Horesh Y, Riabzev M. Fast reed-solomon interactive oracle proofs of proximity. In: Chatzigiannakis I, Kaklamanis C, Marx D, Sannella D, editors. 45th International Colloquium on Automata, Languages, and Programming (ICALP 2018). Vol. 107. Dagstuhl, Germany: Schloss Dagstuhl-Leibniz-Zentrum für Informatik; 2018. p. 14:1–7.
28. Ben-Sasson E, Carmon D, Ishai Y, Kopparty S, Saraf S. Proximity gaps for reed-solomon codes. In: 2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS). Piscataway, NJ, USA: IEEE; 2020. p. 900–9.
29. Arnon G, Chiesa A, Fenzi G, Yogev E. STIR: reed-solomon proximity testing with fewer queries. In: Reyzin L, Stebila D, editors. Advances in Cryptology—CRYPTO 2024. Cham, Switzerland: Springer Nature; 2024. p. 380–413. doi: 10.1007/978-3-031-68403-6_12.
30. Yu M, Sahraei S, Li S, Avestimehr S, Kannan S, Viswanath P. Coded merkle tree: solving data availability attacks in blockchains. In: Bonneau J, Heninger N, editors. Financial cryptography and data security. Cham, Switzerland: Springer International Publishing; 2020. p. 114–34. doi:10.1007/978-3-030-51280-4_8.

31. Sheng P, Xue B, Kannan S, Viswanath P. ACeD: scalable data availability oracle. In: Borisov N, Diaz C, editors. Financial cryptography and data security. Berlin/Heidelberg, Germany: Springer; 2021. p. 299–318. doi:10.1007/978-3-662-64331-0_16.
32. Nazirkhanova K, Neu J, Tse D. Information dispersal with provable retrievability for rollups. In: Proceedings of the 4th ACM Conference on Advances in Financial Technologies, AFT '22. New York, NY, USA: ACM; 2023. p. 180–97. doi:10.1145/3558535.3559778.
33. Wagner B, Zapico A. A documentation of Ethereum's PeerDAS. Cryptology ePrint Archive. 2024.