Notes on Basefold (Part V): IOPP Soundness

- Jade Xie jade@secbit.io
- Yu Guo yu.guo@secbit.io

In this article, we will outline the proof approach for IOPP soundness presented in the [ZCF23] paper, which is similar to the soundness proof for the FRI protocol in [BKS18]. It employs a binary tree method to analyze points where the Prover might cheat, a concept also appearing in the soundness proof of the DEEP-FRI protocol in [BGKS20].

IOPP Protocol

The IOPP protocol has been thoroughly introduced in the second article above. For the analysis in the later sections, we will briefly outline the IOPP protocol here. It is an extension of the FRI protocol, and the process of understanding the protocol can fully leverage the understanding of the FRI protocol, as both the commit and query phases are consistent.

Protocol 1 [ZCF23, Protocol 2] IOPP.commit

Input oracle: $\pi_d \in \mathbb{F}^{n_d}$
Output oracles: $(\pi_{d-1}, \ldots, \pi_0) \in \mathbb{F}^{n_{d-1}} \times \cdots \times \mathbb{F}^{n_0}$

- For i from $d-1$ down to 0:
  1. Verifier samples and sends $\alpha_i \leftarrow \$\mathbb{F}$ from $\mathbb{F}$ to Prover
  2. For each index $j \in [1, n_i]$, Prover: a. Sets $f(X) := \text{interpolate}((\text{diag}(T_i)[j], \pi_{i+1}[j]), (\text{diag}(T_i')[j], \pi_{i+1}[j+n_i]))$ b. Sets $\pi_i[j] = f(\alpha_i)$
  3. Prover outputs oracle $\pi_i \in \mathbb{F}^{n_i}$.

Protocol 2 [ZCF23, Protocol 3] IOPP.query

Input oracles: $(\pi_{d-1}, \ldots, \pi_0) \in \mathbb{F}^{n_{d-1}} \times \ldots \times \mathbb{F}^{n_0}$
Output: accept or reject

- Verifier samples $\mu \leftarrow \$[1, n_{d-1}]$
- For i from $d-1$ down to 0, Verifier:
  1. Queries oracle $\pi_{i+1}[\mu], \pi_{i+1}[\mu + n_i]$
  2. Computes $p(X) := \text{interpolate}((\text{diag}(T_i)[\mu], \pi_{i+1}[\mu]), (\text{diag}(T_i')[\mu], \pi_{i+1}[\mu + n_i]))$
  3. Checks $p(\alpha_i) = \pi_i[\mu]$
  4. If $i > 0$ and $\mu > n_i - 1$, then updates $\mu \leftarrow \mu - n_{i-1}$
- If π_0 is a valid codeword with respect to the generator matrix $\mathbf{G}_0$, output `accept`; otherwise, output `reject`.

Analysis Approach

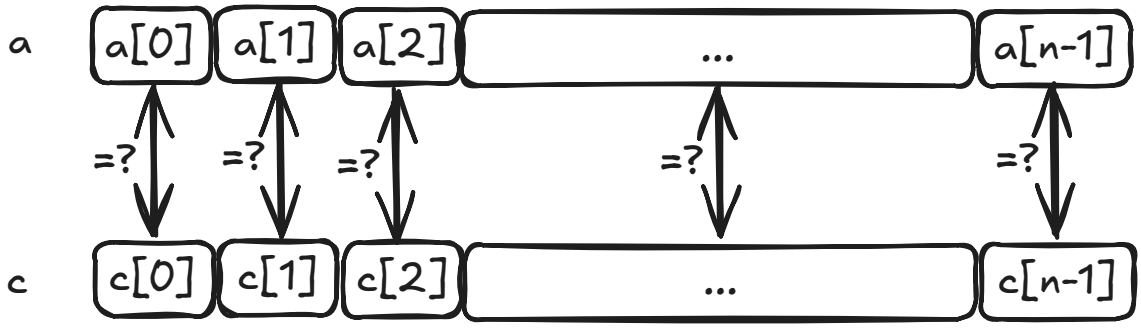
The analysis of IOPP soundness examines, for any Prover who might cheat, what is the maximum probability that the Verifier outputs `accept` in such a scenario. We aim for this probability to be sufficiently small to ensure the protocol's security. Naturally, this probability depends on certain parameters of the protocol, and in practice, we desire it to be below a predetermined security parameter λ (for example, λ could be set to 128 or 256), meaning that this probability should be less than $2^{-\lambda}$.

Let us now examine areas within the IOPP protocol where a cheating Prover might exploit to cause the Verifier to output `accept`. We note that there are two points where the Verifier introduces randomness:

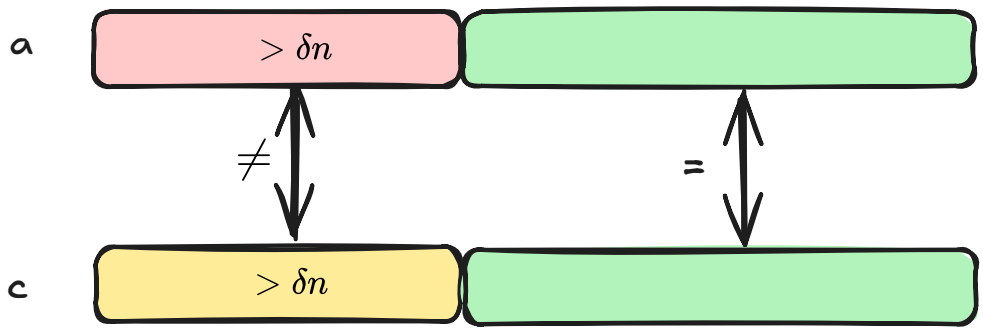
1. In the IOPP.commit phase, step 1 of the protocol, the Verifier selects a random number α_i from $\mathbb{F}$ and sends it to the Prover, who uses it to fold the original π_{i+1} to obtain π_i.
2. In the IOPP.query phase, step 1 of the protocol, the Verifier samples $\mu \leftarrow \$[1, n_{d-1}]$ and then checks whether the Prover's folding was correct.

Suppose the initial cheating Prover provides a π_d that is δ away from C_d. We aim for the Verifier to ultimately check that π_0 is also at least δ away from C_0, meaning that the δ distance is preserved throughout the folding process. Another scenario is detecting that the Prover did not fold correctly. We consider two cases:

1. The Prover is extremely lucky, and the random number α_i chosen by the Verifier causes the folded π_i to be less than δ away from the corresponding C_i, leading the Verifier to output `accept`. We consider this situation very lucky for the Prover because, according to the Proximity Gaps theorem, the probability of such an event is exceedingly small (assuming this probability is ϵ), such that its occurrence is akin to the Prover winning the lottery.
2. The Prover is not as lucky as in Case 1. In this scenario, after folding with the random number, the message π_i still remains at least δ away from the corresponding C_i. Since the Verifier randomly selects $\mu \leftarrow \$[1, n_{d-1}]$ in the IOPP.query phase and only checks a subset of the Prover's foldings, this provides an opportunity for the Prover to potentially evade Verifier checks. For example, if after folding, the message a satisfies $\Delta(a, C) > \delta$ in relative Hamming distance, i.e., $\Delta(a, C) > \delta$. The Verifier will randomly check $a[i]$ against $c[i]$, and if they are unequal, the Verifier will reject.



Since $\Delta(a, C) > \delta$, more than a δ proportion of the components in a differ from the codewords in the encoding space. When the Verifier selects one of these differing positions, it will reject, hence the probability that the Verifier catches the Prover cheating exceeds δ.



If the Verifier queries l times, the probability that the Prover can pass all Verifier checks is at most $(1-\delta)^l$.

Combining the two cases, the probability that the cheating Prover succeeds is bounded above by

$$\epsilon + (1-\delta)^l \tag{1}$$

This represents an overall analysis approach. The specific expression may vary, but the following IOPP soundness theorem will elaborate further.

IOPP Soundness Theorem

Theorem 1 [ZCF23, Theorem 3] (IOPP Soundness for Foldable Linear Codes) Let C_d be a (c, k_0, d) foldable linear code with generator matrices $(G_0, \ldots, G_d)$. Let C_i $(0 \leq i < d)$ denote the code generated by G_i, and assume that for all $i \in [0, d-1]$, the relative minimum distance $\Delta C_i \geq \Delta C_{i+1}$. Let $\gamma > 0$, and set $\delta := \min(\Delta^*(\pi_d, C_d), J_\gamma(J_\gamma(\Delta_{C_d})))$, where $\Delta^*(\pi_d, C_d)$ is the relative coset minimum distance between $\mathbf{v}$ and C_d. Then, for any (adaptively chosen) Prover oracles $\pi_{d-1}, \ldots, \pi_0$, the Verifier in the IOPP.query phase repeated ℓ times outputs `accept` with probability at most $(1-\delta+\gamma d)^\ell$.

The term $J_\gamma(J_\gamma(\Delta_{C_d}))$ mentioned in the theorem refers to the composition of two Johnson functions, defined as follows.

Definition 1 [ZCF23, Definition 4] (Johnson Bound) For any $\gamma \in (0, 1]$, define $J_\gamma : [0,1] \to [0,1]$ as

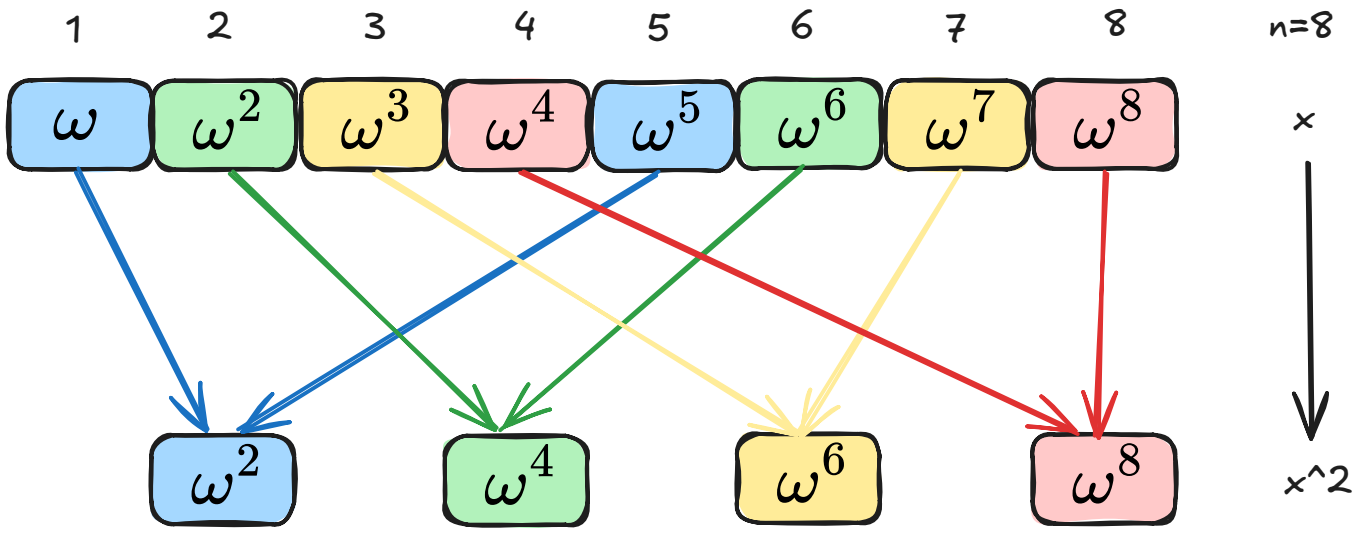
$$J_\gamma(\lambda) := 1 - \sqrt{1 - \lambda(1-\gamma)}. \tag{1}$$

The definition of relative coset minimum distance is as follows.

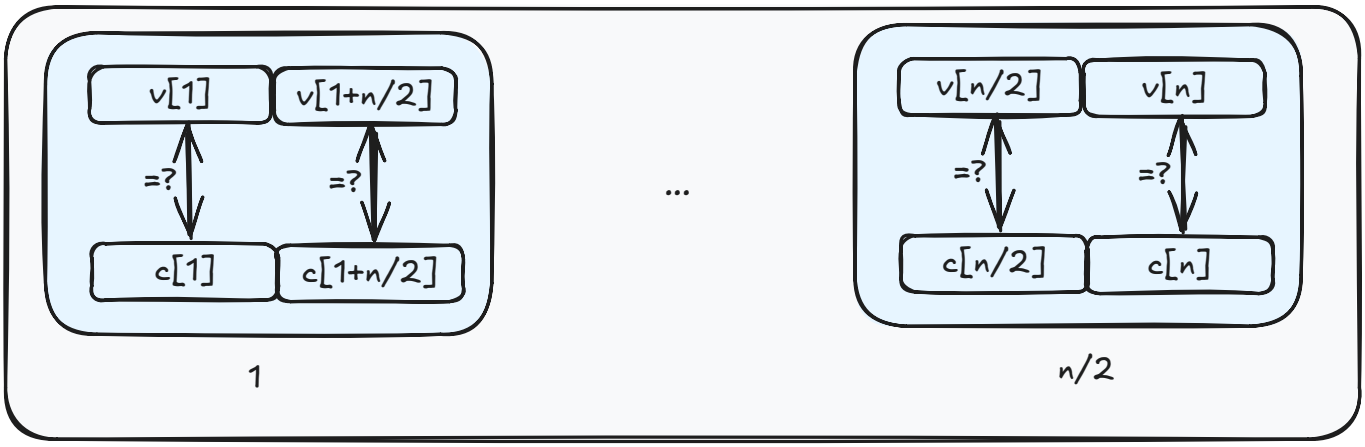
Definition 2 [ZCF23, Definition 5] (Relative Coset Minimum Distance) Let n be an even number, and let C be a $[n, k, d]$ error-correcting code. For a vector $\mathbf{v} \in \mathbb{F}^n$ and a codeword $c \in C$, the relative distance $\Delta^*(\mathbf{v}, c)$ between $\mathbf{v}$ and c is defined as

$$\Delta^*(\mathbf{v}, c) = \frac{2|\{j \in [1, n/2] : \mathbf{v}[j] \neq c[j] \vee \mathbf{v}[j+n/2] \neq c[j+n/2]\}|}{n}. \tag{2}$$

This definition is similar to the block-wise distance definition used in [BBHR18] to prove soundness ([BBHR18, Definition 3.2]). It is an alternative version of the relative minimum Hamming distance. Pairs $\{j, j+n/2\}$ are considered as a pair, corresponding to a coset, analogous to the FRI protocol. For example, for $n = 8$, let the generator be ω with $\omega^8 = 1$, and select the mapping $x \mapsto x^2$, then it can be seen that $\{1, 5\}$, $\{2, 6\}$, $\{3, 7\}$, $\{4, 8\}$ correspond to elements that form a coset, resulting in a total of 4 cosets.



$\Delta^*(\mathbf{v}, c)$ measures the proportion of cosets in which $\mathbf{v}$ and c are not entirely consistent.



Let $\Delta^*(\mathbf{v}, C) := \min_{c \in C} \Delta^*(\mathbf{v}, c)$. Then, it relates to the relative minimum Hamming distance as follows: $\Delta(\mathbf{v}, C) \leq \Delta^*(\mathbf{v}, C)$.

Despite introducing these different definitions and Johnson functions, the proof approach for IOPP soundness remains consistent with the earlier outlined analysis, discussing two cases. Our aim is to analyze the probability that a cheating Prover can pass all Verifier checks and ultimately output `accept`. The proof approach is as follows:

Case 1: The Prover is extremely lucky. Due to the Verifier selecting random numbers α_i, the folded messages are sufficiently close to the encoding space, allowing the Prover to pass all subsequent Verifier checks. For the Verifier, this corresponds to some "bad" events occurring, where there exists an $i \in [0, d-1]$ such that

$$\Delta(\text{fold}_{\alpha_i}(\pi_{i+1}), C_i) \leq \min(\Delta^*(\pi_{i+1}, C_{i+1}), J_\gamma(J_\gamma(\Delta_{C_d}))) - \gamma \tag{3}$$

Using proof by contradiction through the Correlated Agreement theorem (which can derive the corresponding Proximity Gaps theorem), it can be shown that the probability of such "bad" events is small, proven to be at most $\frac{2d}{\gamma^3 |\mathbb{F}|}$.

Case 2: Suppose the Prover is not as lucky, meaning that the "bad" events described in Case 1 do not occur. Then, in the IOPP.query phase, the Verifier selects $\mu \leftarrow \$[1, n_{d-1}]$, and in this scenario, the Prover might evade the Verifier's checks by having the Verifier select points where the Prover has not cheated. Repeating the IOPP.query phase l times, the probability that the Prover manages to pass each check is at most $(1-\delta+\gamma d)^l$.

Combining Cases 1 and 2, for foldable linear codes, the IOPP Soundness is at least

$$\mathbf{s}^{-}(\delta) = 1 - \left(\frac{2d}{\gamma^3 |\mathbb{F}|} + (1-\delta+\gamma d)^l \right). \tag{4}$$

Thus, Theorem 1 is proven.

Proof of Case 1

The following Corollary 1 demonstrates that for any specific i, the probability that after folding, the result is within a relative Hamming distance $\delta - \gamma$ of C_i, denoted as event $B^{(i)}$, is at most $\frac{2}{\gamma^3|\mathbb{F}|}$. Therefore, if certain events B_i occur, their probability does not exceed the sum of the probabilities of these B_i events, i.e.,

$$\Pr\left[\bigcup_{i=0}^{d-1} B^{(i)}\right] \leq \sum_{i=0}^{d-1} \Pr[B^{(i)}] \leq \frac{2d}{\gamma^3|\mathbb{F}|}. \tag{5}$$

Let us examine Corollary 1 in detail.

Corollary 1 [ZCF23, Corollary 1] For any fixed $i \in [0, d-1]$ and $\gamma, \delta > 0$, such that $\delta \leq J_\gamma(J_\gamma(\Delta_{C_d}))$, if $\Delta^*(\mathbf{v}, C_{i+1}) > \delta$, then

$$\Pr_{\alpha_i \leftarrow \$\mathbb{F}}[\Delta(\mathrm{fold}_{\alpha_i}(\mathbf{v}), C_i) \leq \delta - \gamma] \leq \frac{2}{\gamma^3|\mathbb{F}|}. \tag{2}$$

The function $\mathrm{fold}_{\alpha_i}(\cdot)$ is defined as follows. Let $\mathbf{u}, \mathbf{u}' \in \mathbf{F}^{n_i}$ be the unique interpolated vectors such that

$$\pi_{i+1} = (\mathbf{u} + \mathrm{diag}(T_i) \circ \mathbf{u}', \mathbf{u} + \mathrm{diag}(T_i') \circ \mathbf{u}') \tag{6}$$

Then, $\mathrm{fold}_{\alpha_i}(\pi_{i+1})$ is defined as

$$\mathrm{fold}_{\alpha_i}(\pi_{i+1}) := \mathbf{u}' + \alpha_i \mathbf{u}. \tag{7}$$

This essentially represents the process of folding π_{i+1} with the random number α_i.

Corollary 1 generalizes [BKS18] Corollary 7.3 to general foldable linear codes.

Proof Idea of Corollary 1: To prove that the relative Hamming distance after folding with the random number α_i is smaller than the original distance is a rare event, specifically not exceeding $\frac{2}{\gamma^3|\mathbb{F}|}$. Suppose, for contradiction, that this event occurs with a significantly higher probability. Then, by directly applying the Correlated Agreement theorem (from [BKS18] Theorem 4.4), it can be shown that for the affine space $U = \{\mathbf{u} + x\mathbf{u}' : x \in \mathbb{F}\}$, there exists a sufficiently large Correlated Agree subset T within C_i such that there exist $\mathbf{w}, \mathbf{w}' \in C_i$ agreeing with $\mathbf{u}$ and $\mathbf{u}'$ on T, respectively. Encoding $\mathbf{w}, \mathbf{w}'$ yields a codeword c_w in C_{i+1}, thereby estimating $\Delta^*(\mathbf{v}, C_{i+1}) \leq \delta$, which contradicts our assumption. Therefore, the corollary holds.

Proof of Case 2

To prove that repeating the IOPP.query phase l times results in the Verifier outputting `accept` with probability at most $(1-\delta+\gamma d)^l$, we merely need to demonstrate that, in one execution of IOPP.query, the probability that the Verifier outputs `reject` is at least $\delta-\gamma d$.

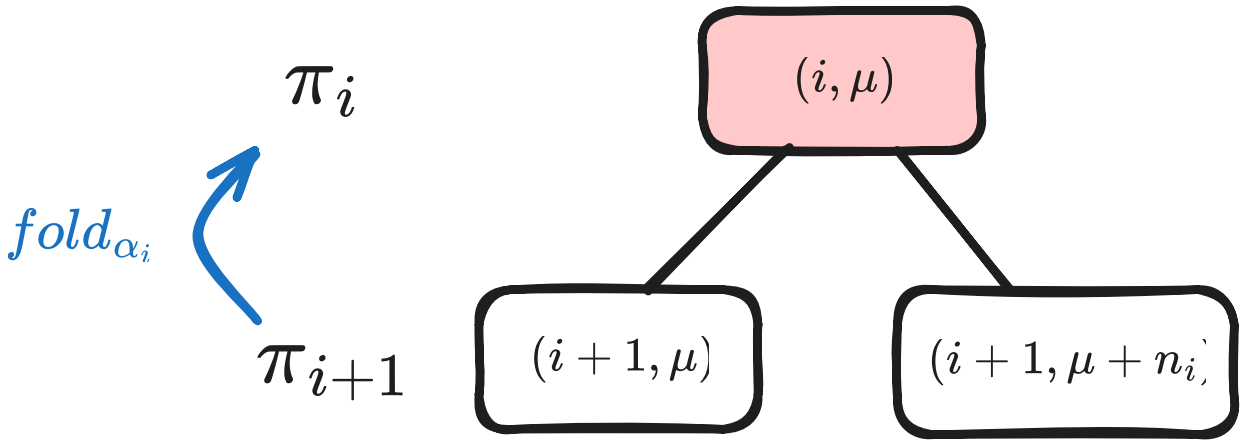
Using the binary tree concept for the proof, we first define a "bad" node (i,μ), as shown in the figure below. These are the points where the Prover fails to pass step 3 of the IOPP.query protocol, meaning that after the Verifier selects a random number μ, for any $i\in[0,d-1]$ and $\mu\in[n_i]$, the Verifier computes in step 2 of IOPP.query:

$$p(X):=\text{interpolate}((\text{diag}(T_i)[\mu],\pi_{i+1}[\mu]),(\text{diag}(T_i')[\mu],\pi_{i+1}[\mu+n_i])) \tag{8}$$

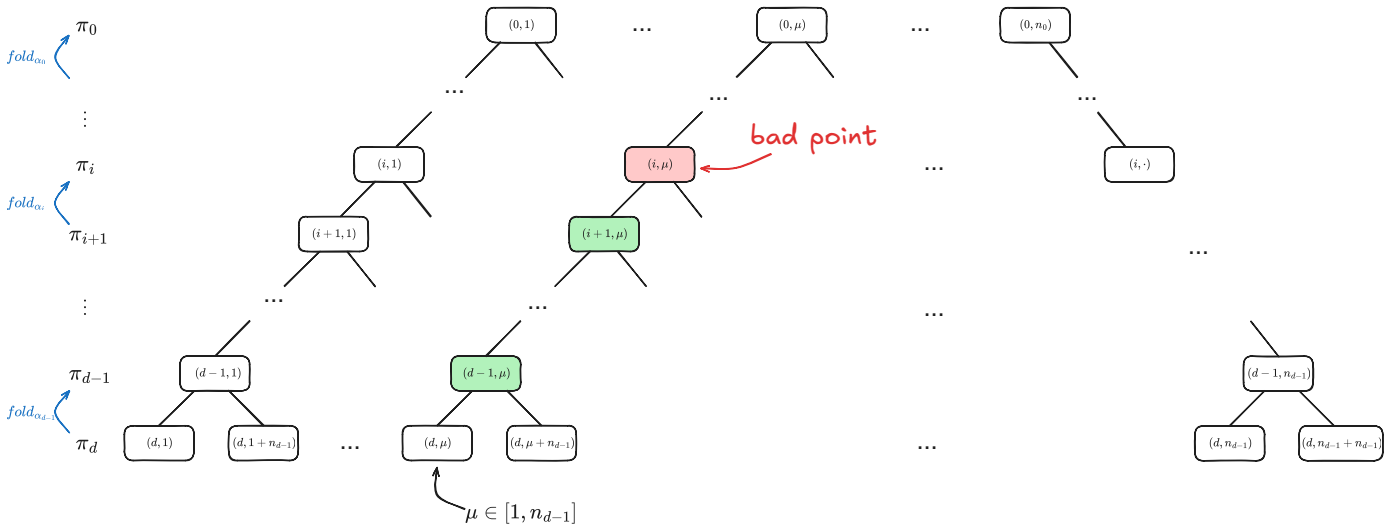
Subsequently, in step 3 of the IOPP.query protocol, the Verifier checks whether

$$p(\alpha_i)\neq\pi_i[\mu] \tag{9}$$

At this point, we say that the node (i,μ) is "bad".



Next, consider i from $d-1$ down to 0. For any $\mu\in[1,n_{d-1}]$, μ can generate a binary tree, forming n_0 such binary trees as depicted below.



If there exists at least one "bad" node (i,μ) within any of these trees—assuming that all nodes from layers $d-1$ down to $i+1$ and their children are "good", i.e., they pass step 3 of the IOPP.query protocol—then when a "bad" node occurs at level i, the Verifier will reject. As shown in the figure, nodes from levels $i+1$ to $d-1$ are all "good". This implies that as long as there is at least one bad node in the entire tree, the Verifier will reject. If we let β_i denote the ratio of "bad" nodes at layer i, then the probability that the Verifier rejects at layer i is β_i. Considering the entire IOPP.query phase, the Verifier's probability of rejection is thus

$\sum_{i=0}^{d-1} \beta_i$, where $\beta_i := \Delta(\pi_i, \text{fold}_{\alpha_i}(\pi_{i+1}))$, representing those "bad" points where the folded π_{i+1} does not agree with π_i.

Thus, the remaining task is to estimate $\sum_{i=0}^{d-1} \beta_i$. [ZCF23, Claim 2] provides inequalities for each β_i.

Claim 1 [ZCF23, Claim 2] For any $i \in [0, d]$, define $\delta^{(i)} := \min(\Delta^*(\pi_i, C_i), J_\gamma(J_\gamma(\Delta_{C_d})))$. For all $i \in [0, d-1]$,

$$\beta_i \geq \delta^{(i+1)} - \delta^{(i)} - \gamma. \tag{10}$$

Under the soundness conditions, $\delta = \delta^{(d)}$. Also, since $\Delta^*(\pi_0, C_0) = \Delta(\pi_0, C_0) = 0$, we have $\delta^{(0)} = 0$. Thus, according to the claim:

$$\delta = \delta^{(d)} - \delta^{(0)} = \sum_{i=0}^{d-1} \delta^{(i+1)} - \delta^{(i)} \leq \sum_{i=0}^{d-1} \beta_i + \gamma d, \tag{11}$$

hence,

$$\sum_{i=0}^{d-1} \beta_i \geq \delta - \gamma d. \tag{12}$$

Therefore, if no bad event B occurs, executing the IOPP.query phase once, the probability that the Verifier rejects is at least $\delta - \gamma d$. This concludes the proof of Case 2.

References

- [BBHR18] Eli Ben-Sasson, Iddo Bentov, Ynon Horesh, and Michael Riabzev. Fast Reed-Solomon Interactive Oracle Proofs of Proximity. In Proceedings of the 45th International Colloquium on Automata, Languages, and Programming (ICALP), 2018. Available online as Report 134-17 on Electronic Colloquium on Computational Complexity.
- [BGKS20] Eli Ben-Sasson, Lior Goldberg, Swastik Kopparty, and Shubhangi Saraf. DEEP-FRI: sampling outside the box improves soundness. In Thomas Vidick, editor, 11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA, volume 151 of LIPIcs, pages 5:1–5:32. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.
- [BKS18] Eli Ben-Sasson, Swastik Kopparty, and Shubhangi Saraf. "Worst-Case to Average Case Reductions for the Distance to a Code". In: Proceedings of the 33rd Computational Complexity Conference. CCC '18. San Diego, California: Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018. ISBN: 9783959770699.
- [ZCF23] Hadas Zeilberger, Binyi Chen, and Ben Fisch. "BaseFold: efficient field-agnostic polynomial commitment schemes from foldable codes." Annual International Cryptology Conference. Cham: Springer Nature Switzerland, 2024.