Basefold-01

Core idea :
- replace MLE quotient-based evaluation arguments (Virgo/Libra/Hyrax) with a Sumcheck + Folding Combo
- Utilize Foldable Linear Codes, which support recursive encoding and folding over arbitrary finite fields.
- instead of relying on typical quotient polynomial approach

$$\frac{f(x) - f(r)}{x - r}$$

Basefold uses : $\Rightarrow$ Sumcheck to reduce MLE evaluations to a random point.
FRI-style folding to recursively prove proximity to a low-degree polynomial

Thus, working better over **non-FFT friendly fields.**

What are Foldable-Linear Codes

let's say we have a linear code

$$C_0 : F_p^{k_0} \rightarrow F_p^{n_0}$$

F_p ✓
$F_p^{k_0} \Rightarrow$ all messages of length k_0 over F_p
$F_p^{n_0} \Rightarrow$ all messages of length n_0 over F_p

A code is a way to map messages to codewords to add redundancy (for error correction/detection)

$\Rightarrow$ C_0 is a function that encodes a message of k_0 elements into a longer vector of n_0 elements

$\Rightarrow$ message $m \in \mathbb{F}_p^{k_0} \xrightarrow{C_0}$ codeword $c \in \mathbb{F}_p^{n_0}$

Redundancy comes from encoding with $n_0 > k_0$

$\Rightarrow R = \frac{n_0}{k_0} \quad \Rightarrow$ Code Rate $= \frac{1}{R}$

now, for every linear code, there exists an encoding matrix $G_0 : \mathbb{F}_p^{k_0 \times n_0}$ such that

$$m\,G_0 = c$$

message, generator matrix, resulting codeword

$m \Rightarrow 1 \times k_0$ $\Rightarrow 1 \times k_0 \times k_0 \times n_0$
$G_0 \Rightarrow k_0 \times n_0$ $= 1 \times n_0$
$\hookrightarrow \underline{\underline{c}}$

Now, basefold wants to build a bigger code C_1 from a base code C_0 such that

$$G_1 = \begin{bmatrix} G_0 & G_0 \\ G_0 . T_0 & G_0 . T_0' \end{bmatrix}$$

$T_0, T_0' \Rightarrow$ diagonal matrices with parameters
↳ they allow recursive folding, like butterfly steps in FFT

→ Now, if we assume G_1 as encoding matrix of another linear code C_1,
↳ parameters of C_1 are $[R, 2k_0, 2n_0]$ ✓ $\Rightarrow$ Code Rate → same
msg. & codeword $\Rightarrow$ doubled length

e.g. $m = (m_0, m_1, m_2, m_3)$; base encoding scheme C_0 is a simple Repetition Code, $k_0 = 2$ & $n_0 = 4$
$\Downarrow$

this means

$\Downarrow$

$k_0 = 2$; So for $m = (m_0, m_1)$

$n_0 = 4$ $\quad C = (m_0, m_1, m_0, m_1)$

Therefore, message length m exactly meets requirements of G_1 i.e.

$k_1 = 2 \cdot k_0 = 4$; $n_1 = 2 \cdot n_0 = 8$

we have $G_0 = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}$ the encoding matrix of base code C_0

assume, T_0 parameters $\Rightarrow (t_0, t_1, t_2, t_3)$

T_0' parameters $\Rightarrow (t_0', t_1', t_2', t_3')$

$$\Rightarrow \quad G_1 = \left[\begin{array}{cccc|cccc} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ \hline t_0 & 0 & t_2 & 0 & t_0' & 0 & t_2' & 0 \\ 0 & t_1 & 0 & t_3 & 0 & t_1' & 0 & t_3' \end{array}\right]$$

$$m G_1 = [m_0 \; m_1 \; m_2 \; m_3] \left[\begin{array}{cccc|cccc} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ \hline t_0 & 0 & t_2 & 0 & t_0' & 0 & t_2' & 0 \\ 0 & t_1 & 0 & t_3 & 0 & t_1' & 0 & t_3' \end{array}\right]$$

$$\Rightarrow \quad \begin{matrix} m_0 + t_0 m_2 & m_1 + t_1 m_3 & m_0 + t_2 m_2 & m_1 + t_3 m_3 \\ m_0 + t_0' m_2 & m_1 + t_1' m_3 & m_0 + t_2' m_2 & m_1 + t_3' m_3 \end{matrix}$$

$\Downarrow$ Let's structurise and make it more clear

Let $m_l = m_0 \; m_1$ $\quad \Rightarrow m G_1 = (m_l \,\|\, m_r) G_1$

$m_r = m_2 \; m_3$

$$= [1 \;\; 1] \begin{bmatrix} m_l & 0 \\ 0 & m_r \end{bmatrix} \begin{bmatrix} G_0 & G_0 \\ G_0 \cdot T_0 & G_0 \cdot T_0' \end{bmatrix}$$

$$= [1 \;\; 1] \begin{bmatrix} m_l G_0 & m_l G_0 \end{bmatrix}$$

$$\left[m_r G_0 . T_0 \quad m_r G_0 T_0' \right]$$

let's see, $m_\ell G_0 = [m_0 \ m_1] \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} = (m_0 \ m_1 \ m_0 \ m_1)$

$$m_r (G_0 T_0) = (t_0 m_2, t_1 m_3, t_2 m_2, t_3 m_3)$$

& similarly $m_r (G_0 T_0') = (t_0' m_2, t_1' m_3, t_2' m_2, t_3' m_3)$

Simplifying, we have

$$mG_1 = m_\ell G_0 + (t_0, t_1, t_2, t_3) \circ m_r G_0 \, \| \, m_\ell G_0 + (t_0', t_1', t_2', t_3') \circ m_r G_0$$

$\Downarrow$

we can notice that the equation resembles butterfly operation in FFT

$\Downarrow$

$$a' = a + t \circ b \qquad b' = a - t \circ b$$

$\Downarrow$

we can recursively construct $G_2, \ldots, G_d$, obtaining a linear code: $C_d: \mathbb{F}_p^k \to \mathbb{F}_p^n$

where $k = k_0 . 2^d$
$n = c_0 k_0 . 2^d$

the code rate is $\rho = \frac{1}{R}$

k_0: msg. length of base encoding

n_0: base encoding length

- We use G_d to denote generator matrix of C_d, $G_d \in F_p^{k\times n}$
- for $i \in \{1,2,\dots,d\}$, we have a recursive relation

$$G_i = \begin{bmatrix} G_{i-1} & G_{i-1} \\ G_{i-1}\cdot T_{i-1} & G_{i-1}\cdot T'_{i-1} \end{bmatrix}$$

here, T_{i-1} and T'_{i-1} are diagonal matrices,

$$T_{i-1} = \begin{bmatrix} t_{i-1,0} & 0 & \cdots & 0 \\ 0 & t_{i-1,1} & & \vdots \\ \vdots & & \ddots & \\ 0 & \cdots & & t_{i-1,n_i-1} \end{bmatrix} \quad T'_{i-1} = \begin{bmatrix} t'_{i-1,0} & 0 & \cdots & 0 \\ 0 & t'_{i-1,1} & & \vdots \\ \vdots & & \ddots & \\ 0 & \cdots & & t'_{i-1,n_i-1} \end{bmatrix}$$

and their diagonal elements at the same positions are distinct:

$$\forall j \in [0, n_i-1], \quad t_{i,j} \neq t'_{i,j}$$

Now, if we have G_d, then for message m_d of length k, the encoding would be

$$w_d = m_d G_d$$

which is inefficient with complexity $O(k\cdot n)$

$\Downarrow$

we can instead use the recursive relation

$\Downarrow$

we split m_d to m_l and m_r

$\Downarrow$

$$w_d = \left(m_l G_{d-1} + \mathrm{diag}(T_d)\circ m_r G_{d-1}\right) \,\|\, \left(m_l G_{d-1} + \mathrm{diag}(T'_d)\circ m_r G_{d-1}\right)$$

$\Downarrow$

splitting until we reach the message length k_0, and then we use G_0 for base encoding

$\Downarrow$

The encoding complexity at this stage would be $O(k_0 n_0)$ and with d recursive rounds, overall complexity becomes $O(n \log(n))$

RBO

000 (0) $\rightarrow$ 000 (0)
001 (1) $\rightarrow$ 100 (4)
010 (2) $\rightarrow$ 010 (2)
011 (3) $\rightarrow$ 110 (6)
100 (4) $\rightarrow$ 001 (1)
101 (5) $\rightarrow$ 101 (5)
110 (6) $\rightarrow$ 011 (3)
111 (7) $\rightarrow$ 111 (7)

$\omega \in F$ is n-th root of unity if $\omega^n = 1$ & $\omega^k \neq 1$ $1 \le k < n$

$\downarrow$

generates all n-th roots of unity

$\hookrightarrow \{1, \omega, \omega^2, \ldots, \omega^{n-1}\}$

- cyclic group under multiplication
- $\omega^{-k} = \omega^{n-k}$ as $\omega^n = 1$
- $1 + \omega + \omega^2 + \cdots + \omega^{n-1} = 0$

Therefore, we go from

this

$$G^{(RS)} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 & \omega^7 \\ 1 & \omega^2 & \omega^4 & \omega^6 & 1 & \omega^2 & \omega^4 & \omega^6 \\ 1 & \omega^3 & \omega^6 & \omega^1 & \omega^4 & \omega^7 & \omega^2 & \omega^5 \\ 1 & \omega^4 & 1 & \omega^4 & 1 & \omega^4 & 1 & \omega^4 \\ 1 & \omega^5 & \omega^2 & \omega^7 & \omega^4 & \omega & \omega^6 & \omega^3 \\ 1 & \omega^6 & \omega^4 & \omega^2 & 1 & \omega^6 & \omega^4 & \omega^2 \\ 1 & \omega^7 & \omega^6 & \omega^5 & \omega^4 & \omega^3 & \omega^2 & \omega \end{bmatrix}$$

$\longrightarrow$ to $\longrightarrow$

this

$$G_2 = \left[\begin{array}{cccc|cccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega^4 & 1 & \omega^4 & 1 & \omega^4 & 1 & \omega^4 \\ 1 & \omega^2 & \omega^4 & \omega^6 & 1 & \omega^2 & \omega^4 & \omega^6 \\ 1 & \omega^6 & \omega^4 & \omega^2 & 1 & \omega^6 & \omega^4 & \omega^2 \\ \hline 1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 & \omega^7 \\ 1 & \omega^5 & \omega^2 & \omega^7 & \omega^4 & \omega & \omega^6 & \omega^3 \\ 1 & \omega^3 & \omega^6 & \omega^1 & \omega^4 & \omega^7 & \omega^2 & \omega^5 \\ 1 & \omega^7 & \omega^6 & \omega^5 & \omega^4 & \omega^3 & \omega^2 & \omega \end{array}\right]$$

$\Downarrow$

$$G_2 = \begin{bmatrix} G_1 & G_1 \\ G_1T_1 & G_1T_1' \end{bmatrix}$$

here

$$T_1 = \begin{bmatrix} 1 & & & \\ & \omega & & \\ & & \omega^2 & \\ & & & \omega^3 \end{bmatrix} \qquad T_1' = \begin{bmatrix} \omega^4 & & & \\ & \omega^5 & & \\ & & \omega^6 & \\ & & & \omega^7 \end{bmatrix}$$

since $\omega^4 \Rightarrow -1$

$\Downarrow$

$T_1' = -T_1$

Similarly $T_2' = -T_2$

we can recursively decompose G_1

$$G_1 = \left[\begin{array}{cc|cc} 1 & 1 & 1 & 1 \\ 1 & \omega^4 & 1 & \omega^4 \\ \hline 1 & \omega^2 & \omega^4 & \omega^6 \\ 1 & \omega^6 & \omega^4 & \omega^2 \end{array}\right] \Rightarrow G_0 = \begin{bmatrix} 1 & 1 \\ 1 & \omega^4 \end{bmatrix}$$

$$T_0 = \begin{bmatrix} 1 & \\ & \omega^2 \end{bmatrix}$$

$$T_0' = \begin{bmatrix} \omega^4 & \\ & \omega^6 \end{bmatrix}$$