Toy single MLE Sumcheck

Karthik Inbasekar[a]

[a] *Ingonyama*

E-mail: karthik@ingonyama.com

Abstract: This is just a rough draft that illustrates the idea of MLE sumcheck complexity. Also this is not entirely new I think. Some form of it probably exists in the literature, but not written out explicitly like this.

1 MLE-sumcheck complexity

1.1 toy example

Consider an execution trace unrolled in the form of a vector, for simplicity we work on a vector of size 8. We interpolate these values using a polynomial $F(X_3, X_2, X_1)$ as follows We interpolate the values using the expression

Eval	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7
Index	000	001	010	011	100	101	110	111
	$F(0,0,0)$	$F(0,0,1)$	$F(0,1,0)$	$F(0,1,1)$	$F(1,0,0)$	$F(1,0,1)$	$F(1,1,0)$	$F(1,1,1)$

Table 1. The elements in the first row are finite field elements, in the second row their index positions are in the boolean representation and the last row represents the interpolation idea.

$$\begin{aligned} F(X_3, X_2, X_1) =& f_0(1-X_3)(1-X_2)(1-X_1) + f_1(1-X_3)(1-X_2)X_1 \\ &+ f_2(1-X_3)X_2(1-X_1) + f_3(1-X_3)X_2X_1) \\ &+ f_4X_3(1-X_2)(1-X_1) + f_5X_3(1-X_2)X_1 \\ &+ f_6X_3X_2(1-X_1) + f_7X_3X_2X_1 \end{aligned} \tag{1.1}$$

Let the sumcheck problem in this case,

$$\sum_{X_i \in \{0,1\}} F(X_3, X_2, X_1) \stackrel{?}{=} C \tag{1.2}$$

proceed with the following steps What we would like to estimate is the complexity of the prover in the evaluation of the polynomials. The commitment of each of the univariate polynomials by the prover is not a bottleneck, since it is a simple EC addition or a very tiny MSM.

Round	Prover $\mathcal{P}$	Communication	Verifier $\mathcal{V}$
1	$r_1(X) := \sum_{X_i \in \{0,1\}} F(X_3, X_2, X)$	$r_1(X) \longrightarrow$ $\longleftarrow \alpha_1 \in \mathbb{F}$	$C \stackrel{?}{=} \sum_{X \in \{0,1\}} r_1(X)$
2	$r_2(X) := \sum_{X_i \in \{0,1\}} F(X_3, X, \alpha_1)$	$r_2(X) \longrightarrow$ $\longleftarrow \alpha_2 \in \mathbb{F}$	$r_1(\alpha_1) \stackrel{?}{=} \sum_{X \in \{0,1\}} r_2(X)$
3	$r_3(X) := F(X, \alpha_2, \alpha_1)$	$r_3(X) \longrightarrow$ $\longleftarrow \alpha_2 \in \mathbb{F}$	$r_2(\alpha_2) \stackrel{?}{=} \sum_{X \in \{0,1\}} r_3(X)$ $r_3(\alpha_3) \stackrel{?}{=} F(\alpha_3, \alpha_2, \alpha_1)$

Table 2. Sumcheck protocol summary

After a short calculation we find that the structure of the polynomial at the end of round 1 is

$$r_1(X) = \sum_{X_i \in \{0,1\}} F(X_2, X_2, X) = \sum_{i=0}^{3} r_1^{(i)}(X) \tag{1.3}$$

$$r_1^{(i)}(X) = f_{2i}(1 - X) + f_{2i+1}X \tag{1.4}$$

The polynomials X and $(1-X)$ are nothing but the Lagrange base polynomials in a binary field. As a consistency check

$$\sum_{X \in \{0,1\}} r_1(X) = r_1(0) + r_1(1) = \sum_{i=0}^{3} (f_{2i} + f_{2i+1}) = C \tag{1.5}$$

Let the challenge at the end of round 1 be $\alpha_1 \in \mathbb{F}_p$ as in table 2. Then the polynomial in round 2 can be written as

$$r_2(X) = \sum_{X_i \in \{0,1\}} F(X_2, X, \alpha_1) = \sum_{i=0}^{1} r_2^{(i)}(X)$$

$$r_2^{(i)}(X) = r_1^{(2i)}(\alpha_1)(1 - X) + r_1^{(2i+1)}(\alpha_1)X \tag{1.6}$$

Similarly let the challenge at the end of round 2 be $\alpha_2 \in \mathbb{F}_p$ as in table 2. Then the polynomial in round 3 can be written as

$$\begin{aligned} r_3(X) &= F(X, \alpha_2, \alpha_1) \\ &= r_2^{(0)}(\alpha_2)(1 - X) + r_2^{(1)}(\alpha_2)X \end{aligned} \tag{1.7}$$

The prover complexity in this case is summarized in table 3

1.2 General form

Consider a execution trace of size $T = 2^n$ (for simplicity we assume that the trace is always zero padded to a power of 2, although this need not be a necessary thing). This trace can be interpolated using a multi variate polynomial of $n = \log_2 T$ variables. We label the trace element with the indexing as in table 1 as $0, 1 \ldots n-1$. Let us represent

Round	Action	Memory	Mult	Add
pre	store f_i, $\forall i = 0, 1, \ldots, 7$	$8\mathbb{F}$	–	–
1	Read f_i ,$\forall i = 0, 1, \ldots, 7$ commit $r_1(X)$ and recieve α_1			
	Read f_i, $\forall i = 0, 1, \ldots, 7$ compute $r_1^{(i)}(\alpha_1)$, $\forall i = 0, 1, 2, 3$	–	8	4
	Clear f_i ,$\forall i = 0, 1, \ldots, 7$ Store $r_1^{(i)}(\alpha_1)$, $\forall i = 0, 1, 2, 3$	$4\mathbb{F}$	–	–
2	Read $r_1^{(i)}(\alpha_1)$, $\forall i = 0, 1, 2, 3$ commit $r_2(X)$ and recieve α_2			
	Read $r_1^{(i)}(\alpha_1)$, $\forall i = 0, 1, 2, 3$ compute $r_2^{(i)}(\alpha_2)$, $\forall i = 0, 1$	–	4	2
	clear $r_1^{(i)}(\alpha_1)$, $\forall i = 0, 1, 2, 3$ store $r_2^{(i)}(\alpha_2)$, $\forall i = 0, 1$	$2\mathbb{F}$	–	–
3	read $r_2^{(i)}(\alpha_2)$, $\forall i = 0, 1$ commit $r_3(X)$			
	clear $r_2^{(i)}(\alpha_2)$, $\forall i = 0, 1$			

Table 3. MLE sumcheck complexity, we have not included the MSM of the three linear polynomials in the above discussion. The notation $x\mathbb{F}$ refers to x finite field elements in memory.

the Boolean representation of the indices with n Boolean variables as $X_1, \ldots X_n$ such that $X_i \in \{0, 1\}$. These n Boolean variables generate what is known as the Boolean hypercube. Thus we represent the interpolated polynomial as $F(X_n, X_{n-1}, \ldots, X_2, X_1)$. For a given trace element the variable X_1 represents the LSB of the Binary representation of the index, while X_n represents the MSB of the Binary representation of the index. The recursive evaluation formula to be executed after committing to polynomial $r_{n-1}(X)$ and obtaining α_{n-1} is derived below.

- At the end of the first round, the prover needs to compute the polynomial

$$r_1(X) = \sum_{X_i \in \{0,1\}} F(X_1, X_2, \ldots, X_{n-1}, X) = \sum_{i=0}^{T/2-1} r_1^{(i)}(X) \tag{1.8}$$

$$r_1^{(i)}(X) = (1 - X) f_{2i} + X f_{2i+1} \tag{1.9}$$

  The prover commits to $r_1(X)$ obtains α_1, and evaluates (Memoization for next round)

$$r_1^{(i)}(\alpha_1) = (1 - \alpha_1) f_{2i} + \alpha_1 f_{2i+1} \ \forall i = 0, 1 \ldots T/2 - 1 \tag{1.10}$$

- At the end of the second round, the prover needs to compute the polynomial

$$r_2(X) = \sum_{X_i \in \{0,1\}} F(X_1, X_2, \ldots, X_{n-2}, X, \alpha_1) = \sum_{i=0}^{T/4-1} r_2^{(i)}(X) \tag{1.11}$$

$$r_2^{(i)}(X) = (1 - X) r_1^{(2i)}(\alpha_1) + X r_1^{(2i+1)}(\alpha_1) \tag{1.12}$$

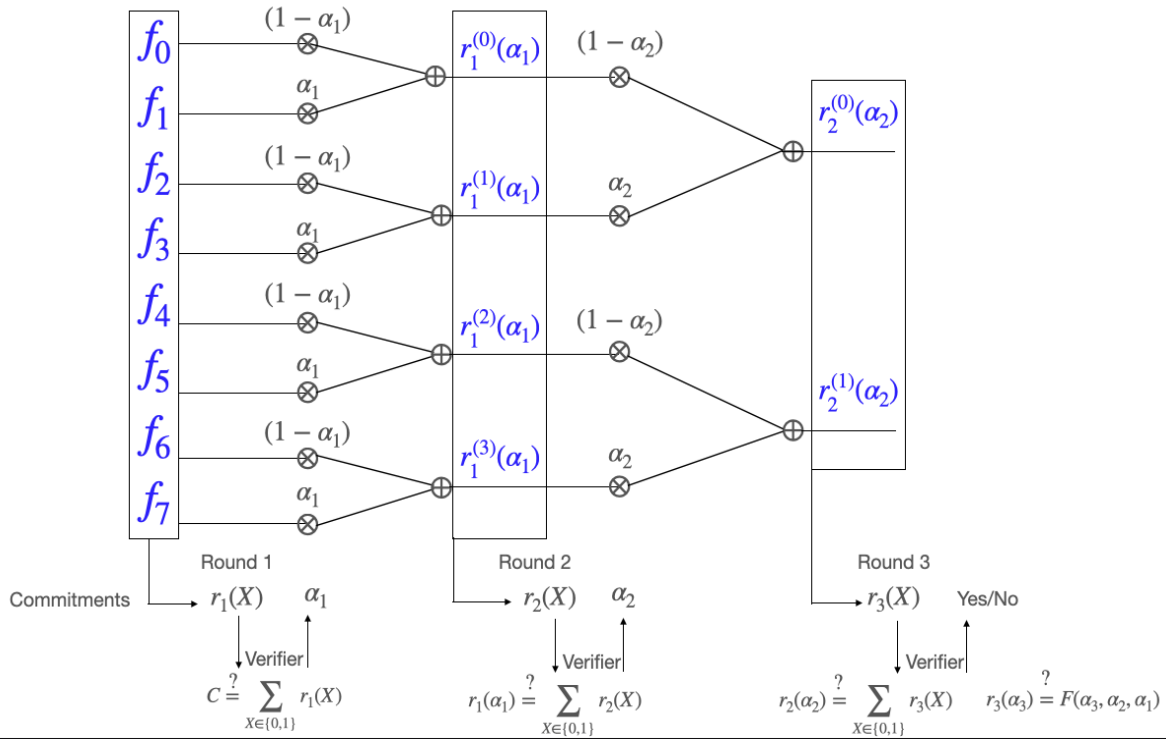

Figure 1. Recursive structure of prover computation in the MLE sumcheck problem. Consider a vector of size T (here $T = 2^3$), One can think of the sumcheck as extending over $n = log_2(T)$ stages. The total number of multiplications are $2(T-2)$ and additions are $T-2$ and thus the complexity is linear in the input size!

The prover commits to $r_2(X)$ obtains α_2, and evaluates (Memoization for next round)

$$r_2^{(i)}(\alpha_2) = (1-\alpha_2)r_1^{(2i)}(\alpha_1) + \alpha_2 r_1^{(2i+1)}(\alpha_1)\ \forall i = 0, 1 \ldots T/4 - 1 \tag{1.13}$$

- Until the $k-1$th round the prover needs to compute the polynomial

$$r_{k-1}(X) = \sum_{X_i\in\{0,1\}} F(X_1, X_2, \ldots, X_{k-2}, X, \ldots, \alpha_2, \alpha_1) = \sum_{i=0}^{\frac{T}{2^{k-1}}-1} r_{k-1}^{(i)}(X)$$

$$r_{k-1}^{(i)}(X) = (1-X)r_{k-2}^{(2i)}(\alpha_{k-2}) + X r_{k-2}^{(2i+1)}(\alpha_{k-2}) \tag{1.14}$$

The prover commits to $r_{k-1}(X)$ obtains α_{k-1}, and evaluates (Memoization for next round)

$$r_{k-1}^{(i)}(\alpha_{k-1}) = (1-\alpha_{k-1})r_{k-2}^{(2i)}(\alpha_{k-2}) + \alpha_{k-1} r_{k-2}^{(2i+1)}(\alpha_{k-2})\ \forall i = 0, 1 \ldots T/2^{k-1} - 1 \tag{1.15}$$

- In the final round $n = \log_2(T)$, the prover computes the polynomial

$$r_n(X) = (1-X)r_{n-1}^{(0)}(\alpha_{n-1}) + X r_{n-1}^{(1)}(\alpha_{n-1}) \tag{1.16}$$

and commits to it.

The Prover complexity of the protocol is summarized in table 4. The computation proceeds in $n = \log_2 T$ stages, and in the kth round ($k = 1, 2, \ldots$), there are $T/2^{k-1}$ multiplications and $T/2^k$ additions, with a memory of $T/2^k$ required to be stored for the $k+1$th round. It can be checked that the total number of multiplications is $2T - 4$ and total number of additions are $T - 2$ being linear in the size of the inputs. The reduction of logarithmic to linear multiplications is the big performance gain of the MLE sumcheck protocol.

In our way of characterizing, the $k-1$th round requires 2 reads of vectors of size $T/2^{k-1}$ and one write of vector of size $T/2^k$, which will be the bottomline for latency in the MLE sumcheck. One more point which we have glossed over is in the MSM computation. The odd and even coefficients of every vector needs to be summed before committing the final linear polynomial, but we think field additions are not much of a bottleneck and ignore this subtlety.

Round	Action	Memory	Mult	Add
pre	store f_i, $\forall i = 0, 1, \ldots, T-1$	$T\mathbb{F}$	–	–
1	Read f_i ,$\forall i = 0, 1, \ldots, T-1$ commit $r_1(X)$ and recieve α_1			
	Read f_i, $\forall i = 0, 1, \ldots, T-1$ compute $r_1^{(i)}(\alpha_1)$, $\forall i = 0, 1, \ldots, T/2-1$	–	T	$T/2$
	Clear f_i ,$\forall i = 0, 1, \ldots, T-1$ Store $r_1^{(i)}(\alpha_1)$, $\forall i = 0, 1, \ldots, T/2-1$	$\frac{T}{2}\mathbb{F}$	–	–
2	Read $r_1^{(i)}(\alpha_1)$, $\forall i = 0, 1, \ldots, T/2-1$ commit $r_2(X)$ and recieve α_2			
	Read $r_1^{(i)}(\alpha_1)$, $\forall i = 0, 1, \ldots T/2-1$ compute $r_2^{(i)}(\alpha_2)$, $\forall i = 0, 1, \ldots T/4-1$	–	$T/2$	$T/4$
	clear $r_1^{(i)}(\alpha_1)$, $\forall i = 0, 1, \ldots T/2-1$ store $r_2^{(i)}(\alpha_2)$, $\forall i = 0, 1, \ldots T/4-1$	$\frac{T}{4}\mathbb{F}$	–	–
$\vdots$	$\vdots$	$\vdots$	$\vdots$	
$k-1$	Read $r_{k-2}^{(i)}(\alpha_{k-2})$, $\forall i = 0, 1, \ldots T/2^{k-1}-1$ commit $r_{k-1}(X)$ and recieve α_{k-1}			
	Read $r_{k-2}^{(i)}(\alpha_{k-2})$, $\forall i = 0, 1, \ldots T/2^{k-1}-1$ compute $r_{k-1}^{(i)}(\alpha_{k-1})$, $\forall i = 0, 1, \ldots T/2^{k}-1$	–	$T/2^{k-1}$	$T/2^k$
	clear $r_{k-2}^{(i)}(\alpha_{k-2})$, $\forall i = 0, 1, \ldots T/2^{k-1}-1$ store $r_{k-1}^{(i)}(\alpha_{k-1})$, $\forall i = 0, 1, \ldots T/2^{k}-1$	$\frac{T}{2^k}\mathbb{F}$	–	–
$\vdots$	$\vdots$	$\vdots$	$\vdots$	
$n = \log_2 T$	Read $r_{n-1}^{(i)}(\alpha_{n-1})$, $\forall i = 0, 1$ commit $r_n(X)$			

Table 4. Prover MLE sumcheck complexity, we have not included the MSM of the three linear polynomials in the above discussion. The notation $x\mathbb{F}$ refers to x finite field elements in memory.

Acknowledgments

We stand on the shoulders of giants.