GKR

$S=7 \qquad d=3$

circuit C of size S, depth d, fan-in 2

$\tilde{W}_i \to$ MLE of W_i

$W_i : \{0,1\}^{k_i} \to F$

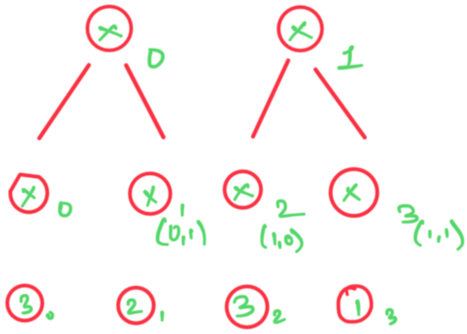
$\tilde{W}_i$

$$\boxed{\begin{array}{l} in_{1,i}, in_{2,i} : \{0,1\}^{k_i} \to \{0,1\}^{k_{i+1}} \\ \text{take i/p label } a \text{ of a gate at layer } i \\ \text{o/p label of gate b \& c in layer } i+1 \end{array}}$$

$i=0 \quad S_0=2$ layer 0
$k_0=1$

$i=1 \quad S_1=4$ layer 1
$k_1=2$

$i=2 \quad S_2=4$ layers 2
$k_2=2$



$add_i : \{0,1\}^{k_i+2k_{i+1}} \to \{0,1\} \qquad \tilde{add}_i$

$mult_i : \{0,1\}^{k_i+2k_{i+1}} \to \{0,1\} \qquad \tilde{mult}_i$

$add_i(a,b,c)=1$ iff $(b,c)=(in_{1,i}(a), in_{2,i}(a))$

for each layer i, add_i, $mult_i$ depend only on the circuit C and not on the input x to C

in contrast the function W_i does depend on x
↳ cause W_i maps each gate label at layer i to the value of the gate when C is evaluated on input x

Detailed description

- d iterations, one for each layer of the circuit
- each iteration i starts with P claiming a value for $\tilde{W}_i(r_i)$ for some point $r_i \in F^{k_i}$

$\left(\begin{array}{l} \text{say } i=1 \Rightarrow S_1=4 \Rightarrow k_1=2 \\ \Rightarrow \tilde{W}_1(r_1) \qquad r_1 \in F^2 \end{array} \right)$

$S_0 = 2^{k_0} \qquad S_0 = 2 \qquad k_0 = 1 \qquad D : \{0,1\}^1 \to F$ maps label of an o/p gate to claimed value of o/p

verifier picks
$r_0 \in F^{k_0}$, $r_1 \in F^1$

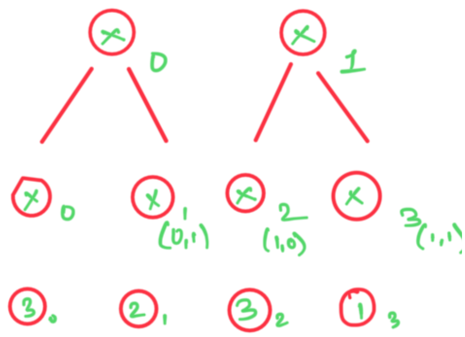
eval. $\tilde{D}(r_0)$

if $\tilde{D}(r_0) = \tilde{W}_0(r_0)$

$i=0$ $S_0=2$ layer 0
$k_0=1$

$i=1$ $S_1=4$ layer 1
$k_1=2$

$i=2$ $S_2=4$ layer 2
$k_2=2$



The purpose of iteration i is to reduce the claim about the value of $\widetilde{W}_i(r_i)$ to a claim about $\widetilde{W}_{i+1}(r_{i+1})$ for some $r_{i+1} \in \mathbb{F}^{k_{i+1}}$, in the sense that it is safe for $\mathcal{V}$ to assume that the first claim is true as long as the second claim is true. To accomplish this, the iteration applies the sum-check protocol to a specific polynomial derived from $\widetilde{W}_{i+1}$, $\widetilde{\text{add}}_i$, and $\widetilde{\text{mult}}_i$. Our description of the protocol actually makes use of a simplification due to Thaler [Tha15].

$\Rightarrow$ reduce claim from
$\tilde{W}_0(r_0) \to \tilde{W}_1(r_1)$
$r_1 \in F^{k_1} \Rightarrow r_1 \in F^2$

sumcheck on polynomial derived
from $\tilde{W}_{i+1}$, $\tilde{add}_i$ and $\tilde{mult}_i$

$$\tilde{W}_i(z) = \sum_{b,c \in \{0,1\}^{k_{i+1}}} \underbrace{\tilde{add}_i(z,b,c)\left(\tilde{W}_{i+1}(b) + \tilde{W}_{i+1}(c)\right) + \tilde{mult}_i(z,b,c)\left(\tilde{W}_{i+1}(b) \cdot \tilde{W}_{i+1}(c)\right)}_{\alpha}$$

LHS and RHS agree for all $a \in \{0,1\}^{k_i}$ ✓

proof since,

$$add_i(a,b,c) = \begin{cases} 1 & \text{if } (b,c) = (in_1(a), in_2(a)) \\ 0 & \text{otherwise} \end{cases}$$

similarly for $mult_i(a,b,c)$ $\forall (b,c) \in \{0,1\}^{k_{i+1}}$

$$\Rightarrow \alpha = \tilde{W}_{i+1}(in_1(a)) + \tilde{W}_{i+1}(in_2(a)) = W_{i+1}(in_1(a)) + W_{i+1}(in_2(a))$$
$$= W_i(a)$$
$$= \tilde{W}_i(a)$$

Therefore, in order to check the prover's claim about $\tilde{W}_i(r_i)$,
verifier applies sum-check protocol to

$$f^{(i)}_{r_i}(b,c) = \widetilde{add}_i(r_i,b,c)\left(\tilde{W}_{i+1}(b)+\tilde{W}_{i+1}(c)\right) + \widetilde{mult}_i(r_i,b,c)\cdot \tilde{W}_{i+1}(b)\cdot\tilde{W}_{i+1}(c)$$

$$i=0 \Rightarrow f^{(0)}_{r_0}(b,c) = \widetilde{add}_0(r_0,b,c)\left(\tilde{W}_1(b)+\tilde{W}_1(c)\right) + \widetilde{mult}_0(r_0,b,c)\cdot\tilde{W}_1(b)\tilde{W}_1(c)$$

$r_0 \in F^{k_0}$ $k_0 = 1 \Rightarrow r_0$ can be 0 or 1 ✓ i.e. evaluating $\tilde{W}_0$ at any one point

Note that *the verifier does not know the polynomial* $\tilde{W}_{i+1}$ (as this polynomial is defined in terms of gate values at layer $i+1$ of the circuit, and unless $i+1$ is the input layer, the verifier does not have direct access to the values of these gates), and hence the verifier does not actually know the polynomial $f^{(i)}_{r_i}$ that it is applying the sum-check protocol to. Nonetheless, it is possible for the verifier to apply the sum-check protocol to $f^{(i)}_{r_i}$ because, until the final round, the sum-check protocol does not require the verifier to know anything about the polynomial other than its degree in each variable (see Remark 4.2). However, there remains the issue that $\mathcal{V}$ can only execute the final check in the sum-check protocol if she can evaluate the polynomial $f^{(i)}_{r_i}$ at a random point. This is handled as follows.

Let us denote the random point at which $\mathcal{V}$ must evaluate $f^{(i)}_{r_i}$ by (b^*, c^*) where $b^* \in F^{k_{i+1}}$ are the first k_{i+1} entries and $c^* \in F^{k_{i+1}}$ are the last k_{i+1} entries

$i=0$
$b^* \in F^{k_1}$ $k_1 = 2$
$c^* \in F^{k_1}$

Note that b^* and c^* may have non-Boolean entries ✓

evaluating $f^{(i)}_{r_i}(b^*,c^*)$ requires evaluating $\widetilde{add}_i(r_i,b^*,c^*)$, $\widetilde{mult}_i(r_i,b^*,c^*)$, $\tilde{W}_{i+1}(b^*)$, and $\tilde{W}_{i+1}(c^*)$

For many circuits, particularly those whose wiring pattern displays repeated structure, $\mathcal{V}$ can evaluate $\widetilde{\text{add}}_i(r_i,b^*,c^*)$ and $\widetilde{\text{mult}}_i(r_i,b^*,c^*)$ on her own in $O(k_i+k_{i+1})$ time as well. For now, assume that $\mathcal{V}$ can indeed perform this evaluation in $\text{poly}(k_i,k_{i+1})$ time, but this issue will be discussed further in Section 4.6.6.

$\mathcal{V}$ cannot however evaluate $\tilde{W}_{i+1}(b^*)$ and $\tilde{W}_{i+1}(c^*)$

↳ instead $\mathcal{V}$ asks $\mathcal{P}$ to simply provide values, say, z_1 and z_2, and uses iteration $i+1$ to verify that these values are as claimed.

<u>Complication</u>: the <u>precondition</u> for iteration $i+1$ is that $\mathcal{P}$ claims a value for $\tilde{W}_{i+1}(r_{i+1})$ for a single point $r_{i+1} \in F^{k_{i+1}}$

(precondition for iter. 1 is that $\mathcal{P}$ claims a value for $\tilde{W}_1(r_1)$)

(for a single pt. $r_1 \in F^{k_1}$ $(k_1 = 2)$)

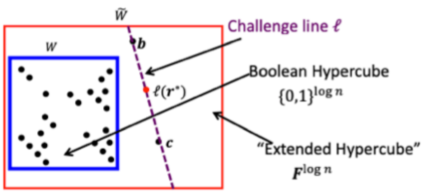
so V needs to reduce verifying both $\tilde{W}_{i+1}(b^*) = z_1$ and $\tilde{W}_{i+1}(c^*) = z_2$ to verifying $\tilde{W}_{i+1}(r_{i+1})$ at a single point $r_{i+1} \in F^{k_{i+1}}$ in the sense that it is safe for V to accept the claimed values $\tilde{W}_{i+1}(b^*)$ and $\tilde{W}_{i+1}(c^*)$ as long as the value of $\tilde{W}_{i+1}(r_{i+1})$ is as claimed.

Reducing to Verification of a Single Point

let $\ell : F \to F^{k_{i+1}}$ be the unique line such that $\ell(0) = b^*$ and $\ell(1) = c^*$

P sends a univariate polynomial q of degree at most k_{i+1} that is claimed to be $\tilde{W}_{i+1} \circ \ell$ (restriction of $\tilde{W}_{i+1}$ to the line ℓ)

$i = 0$
$\Rightarrow k_1 = 2$
$\Rightarrow$ at most 2



V checks that $q(0) = z_1$ and $q(1) = z_2$

picks a random point $r^* \in F$, asks P to prove that $\tilde{W}_{i+1}(\ell(r^*)) = q(r^*)$

as long as V is convinced that $\tilde{W}_{i+1}(\ell(r^*)) = q(r^*)$

↳ it's safe for V to believe that q does in fact equal $\tilde{W}_{i+1} \circ \ell$

$\Downarrow$

$\tilde{w}_{i+1}(b^*) = z_1$ & $\tilde{w}_{i+1}(c^*) = z_2$

as claimed by P

$\downarrow$

Iteration i gets completed; P and V then move on to the iteration for layer i+1 of the circuit, whose purpose is to verify that $\tilde{w}_{i+1}(r_{i+1})$ has the claimed value, where $r_{i+1} := \ell(r^*)$

$\downarrow$

The Final Iteration. At the final iteration d, $\mathcal{V}$ must evaluate $\widetilde{W}_d(r_d)$ on her own. But the vector of gate values at layer d of C is simply the input x to C. By Lemma 3.8, $\mathcal{V}$ can compute $\widetilde{W}_d(r_d)$ on her own in $O(n)$ time, where recall that n is the size of the input x to C.