eryx zkCity 2024

Día 3: Wirings de Plonk

¿Qué vamos a ver hoy?

- Repaso de ayer
- Permutaciones
- Máscaras
- ZK Adventures 3





Programa

| _ • | , | • | |
|------------|-----|--------------|------|
| Ηı | eci | ICIO | nes |
| – J | | <i>4</i> 010 | 1100 |

| Q_L | Q_R | Q_M | Q_O | Q_C |
|-------|-------|-------|-------|-------|
| 0 | 0 | 1 | -1 | 0 |
| 1 | 1 | 0 | -1 | 0 |
| 1 | 0 | 0 | -1 | -1 |



| A | В | С |
|---|---|---|
| 2 | 3 | 6 |
| 6 | 3 | 9 |
| 9 | - | 8 |

XOR

Inputs: x, y Output: z

| \boldsymbol{x} | y | z |
|------------------|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 0 |

XOR

Inputs: x, y Output: z

$$\left\{egin{array}{l} x(1-x)=0\ y(1-y)=0\ z(1-z)=0\ z=x+y-2xy \end{array}
ight.$$

| \boldsymbol{x} | y | z |
|------------------|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 0 |

XOR à la Plonk

Programa

| Q_L | Q_R | Q_M | Q_O | Q_C |
|-------|-------|-------|-------|-------|
| 1 | 0 | -1 | 0 | 0 |
| 1 | 0 | -1 | 0 | 0 |
| 1 | 0 | -1 | 0 | 0 |
| 1 | 1 | -2 | -1 | 0 |



Ejecuciones

| Α | В | С |
|---|---|---|
| 1 | 1 | - |
| 1 | 1 | - |
| 0 | 0 | - |
| 1 | 1 | 0 |

| Α | В | С |
|---|----|---|
| 0 | 1 | - |
| 0 | 0 | - |
| 1 | -1 | - |
| 1 | 1 | 0 |

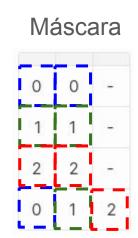




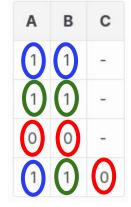
XOR à la Plonk

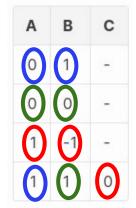
Programa

| Q_L | Q_R | Q_M | Q_O | Q_C |
|-------|-------|-------|-------|-------|
| 1 | 0 | -1 | 0 | 0 |
| 1 | 0 | -1 | 0 | 0 |
| 1 | 0 | -1 | 0 | 0 |
| 1 | 1 | -2 | -1 | 0 |



Ejecuciones









El caminito de hoy

1. Protocolo para

"demostrar que un vector privado es un reordenamiento de otro público"

2. Protocolo para

"demostrar que un vector privado es un reordenamiento de otro privado"

3. Protocolo para

"demostrar que un vector privado respeta una máscara pública"



Datos públicos:

Un vector V con coeficientes en Fp de largo N

Prover:

- Tiene un vector W con coeficientes en Fp de largo N.
- Le envía un oráculo [w] a un verifier
- Trata de convencerlo de que W es un reordenamiento de V

```
# Public vector of field elements
V = [49, 56, 7, 15, 56, 56, 56, 49, 7, 49, 15, 15, 56, 15, 7, 15]
# Private vector of field elements known only to the prover
W = [15, 15, 7, 56, 15, 15, 56, 56, 7, 56, 56, 49, 49, 15, 49, 7]
```

```
def are_permutations(vec1, vec2):
    if len(vec1) != len(vec2):
        return False
    return sorted(vec1) == sorted(vec2)
```

```
# Public vector of field elements
V = [49, 56, 7, 15, 56, 56, 56, 49, 7, 49, 15, 15, 56, 15, 7, 15]
# Private vector of field elements known only to the prover
W = [15, 15, 7, 56, 15, 15, 56, 56, 7, 56, 56, 49, 49, 15, 49, 7]
```

Otro algoritmo?

```
def are_permutations(V, W):
    if len(V) != len(W):
        return False
    product1 = product2 = 1
    for v, w in zip(V, W):
        product1 = product1 * v
        product2 = product2 * w
    return product1 == product2
```

```
# Public vector of field elements
V = [49, 56, 7, 15, 56, 56, 56, 49, 7, 49, 15, 15, 56, 15, 7, 15]
# Private vector of field elements known only to the prover
W = [15, 15, 7, 56, 15, 15, 56, 56, 7, 56, 56, 49, 49, 15, 49, 7]
```

Otro algoritmo

```
def are_permutations(V, W, prime):
    if len(V) != len(W):
        return False
    product1 = product2 = 1
    alpha = random.randint(0, prime - 1)
    for v, w in zip(V, W):
        product1 = product1 * (v + alpha)
        product2 = product2 * (w + alpha)
    return product1 == product2
```

```
# Public vector of field elements
V = [49, 56, 7, 15, 56, 56, 56, 49, 7, 49, 15, 15, 56, 15, 7, 15]
# Private vector of field elements known only to the prover
W = [15, 15, 7, 56, 15, 15, 56, 56, 7, 56, 56, 49, 49, 15, 49, 7]
```

Otro algoritmo

```
def are_permutations(V, W, prime):
    if len(V) != len(W):
        return False
    product = 1
    alpha = random.randint(0, prime - 1)
    for v, w in zip(V, W):
        product = product * (v + alpha) / (w + alpha)
    return product == 1
```

Afirmación: El vector W es una permutación del vector V con alta probabilidad si: para un α aleatorio, existe un polinomio Z tal que:

$$Z(\omega^i)\cdot (v_i+lpha)=Z(\omega^{i+1})\cdot (w_i+lpha)$$
 para todo $i=0,\dots,N-1$.

Afirmación: El vector W es una permutación del vector V con alta probabilidad si: para un α aleatorio, existe un polinomio Z tal que:

1.
$$Z(\omega^i) \cdot (v_i + \alpha) = Z(\omega^{i+1}) \cdot (w_i + \alpha)$$
 para todo $i = 0, \ldots, N-1$.

2.
$$Z(1) \neq 0$$
.

Afirmación: El vector W es una permutación del vector V con alta probabilidad si: para un α aleatorio, existe un polinomio Z tal que

$$Z(1) \neq 0$$

y el polinomio

$$f(X) := Z(X)(v(X) + \alpha) - Z(\omega X)(w(X) + \alpha)$$

cumple
$$f(d) = 0$$
 para todo $d \in D$

Prover Interpola W sobre D Obtiene w Envía un oráculo $\lceil w \rceil$ Construye Z usando a Calcula $t = \frac{Z(X)(v+\alpha) - Z(\omega X)(w+\alpha)}{X^N - 1}$ Envía oráculos [Z] y [t]

Verifier
$$\begin{aligned} &\text{Recibe } [w] \\ &\text{Sortea coeficiente aleatorio } \alpha \\ &\text{Env\'ia } \alpha \text{ al Prover} \\ &\text{Recibe } [\textbf{Z}] \text{ y [t]. Elige } \zeta \text{ random.} \\ &\text{Calcula} \quad a := Z(\zeta) \\ & b := v(\zeta) + \alpha \\ & c := Z(\zeta\omega) \\ & d := w(\zeta) + \alpha \\ & e := t(\zeta) \cdot (\zeta^N - 1) \end{aligned}$$

Verifica ab -cd = e Verifica Z(1)!= 0

Prover Interpola V_1 sobre D y obtiene v_1 Interpola V_2 sobre D y obtiene v_2 Envía oráculos $[v_1]$ y $[v_2]$

Construye Z usando α Calcula

$$t = \frac{Z(X)(v_1 + \alpha) - Z(\omega X)(v_2 + \alpha)}{X^N - 1}$$

Envía oráculos [Z] y [t]



Recibe $[v_1]$ y $[v_2]$

Sortea coeficiente aleatorio a

Envía α al Prover

Recibe [Z] y [t]. Elige ζ random.

Calcula $a := Z(\zeta)$

$$b := v_1(\zeta) + \alpha$$

$$c := Z(\zeta \omega)$$

$$d := v_2(\zeta) + \alpha$$

$$e := t(\zeta) \cdot (\zeta^N - 1)$$

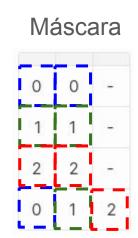
Verifica ab -cd = e Verifica Z(1)!= 0



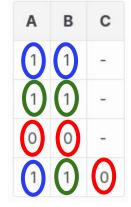
XOR à la Plonk

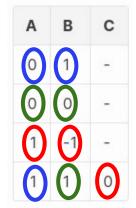
Programa

| Q_L | Q_R | Q_M | Q_O | Q_C |
|-------|-------|-------|-------|-------|
| 1 | 0 | -1 | 0 | 0 |
| 1 | 0 | -1 | 0 | 0 |
| 1 | 0 | -1 | 0 | 0 |
| 1 | 1 | -2 | -1 | 0 |



Ejecuciones









Apuntamos a entender la siguiente frase

El vector W respeta la máscara M con alta probabilidad si el vector de pares

$$((0, w_0), (1, w_1), (2, w_2), \ldots, (n, w_n))$$

es una permutación del vector de pares

$$((\sigma(0), w_0), (\sigma(1), w_1), (\sigma(2), w_2), \dots, (\sigma(n), w_n))$$

| 1 | 0 | 2 |
|---|---|---|
| 2 | 1 | 4 |
| 2 | 4 | 5 |

Vector aplanado

| 1 | 0 2 | | 2 | 1 | 4 | 2 | 4 | 5 |
|---|-----|--|---|---|---|---|---|---|
|---|-----|--|---|---|---|---|---|---|

Permutación

| 4 | 1 | 3 | 6 | 0 | 7 | 2 | 5 | 8 |
|---|---|---|---|---|---|---|---|---|
|---|---|---|---|---|---|---|---|---|

| 1 | - | 2 |
|---|---|---|
| 2 | 1 | 4 |
| 2 | 4 | 5 |

Máscara aplanada

| 1 | 0 | 2 | 2 | 1 | 4 | 2 | 4 | 5 | |
|---|---|---|---|---|---|---|---|---|--|
|---|---|---|---|---|---|---|---|---|--|

| 1 | - | 2 |
|---|---|---|
| 2 | 1 | 4 |
| 2 | 4 | 5 |

Máscara aplanada

| 1 | 0 | 2 | 2 | 1 | 4 | 2 | 4 | 5 | |
|---|---|---|---|---|---|---|---|---|--|
|---|---|---|---|---|---|---|---|---|--|

| 1 | - | 2 |
|---|---|---|
| 2 | 1 | 4 |
| 2 | 4 | 5 |

Máscara aplanada

| 1 | 0 | 2 | 2 | 1 | 4 | 2 | 4 | 5 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |

Permutación

$$\sigma(0) = 4$$

 $\sigma(4) = 0$

$$\sigma(1) = 1$$

$$\sigma(2) = 3$$

 $\sigma(3) = 6$
 $\sigma(6) = 2$

$$\sigma(5) = 7$$

 $\sigma(7) = 5$

$$o(8) = 8$$

| Α | В | С |
|-----|----|------|
| 20 | - | 100 |
| 100 | 20 | 8 |
| 100 | 8 | 1000 |

| (0,20) | (1,0) | (2,100) | (3,100) | (4,20) | (5,8) | (6,100) | (7,8) | (8,1000) |
|--------|-------|---------|-----------------|-----------------|-------|-----------------|---------------|----------|
| 4 | 1 | 3 | 6 | 0 | 7 | 2 | 5 | 8 |
| (4,20) | (1,0) | (3,100) | (6 ,100) | (0 ,20) | (7,8) | (2 ,100) | (5 ,8) | (8,1000) |

Máscara aplanada

1 0 2 2 1 4 5

Permutación

$$\sigma(0) = 4$$

$$\sigma(4) = 0$$

$$\sigma(1) = 1$$

$$\sigma(2) = 3$$

 $\sigma(3) = 6$
 $\sigma(6) = 2$

$$\sigma(5) = 7$$

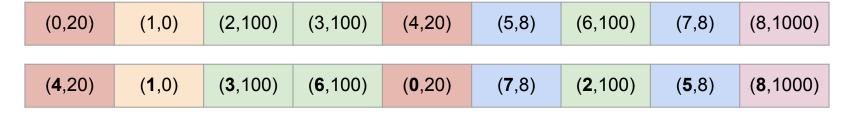
 $\sigma(7) = 5$

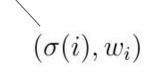
$$\sigma(8) = 8$$

Vector W que satisface la máscara

| | 20 | 0 | 100 | 100 | 20 | 8 | 100 | 8 | 1000 |
|--|----|---|-----|-----|----|---|-----|---|------|
|--|----|---|-----|-----|----|---|-----|---|------|







En general

El vector (aplanado) W respeta la máscara M con alta probabilidad si el vector de pares

$$((0, w_0), (1, w_1), (2, w_2), \ldots, (n, w_n))$$

es una permutación del vector de pares

$$((\sigma(0), w_0), (\sigma(1), w_1), (\sigma(2), w_2), \dots, (\sigma(n), w_n))$$

Problema: Los vectores no son vectores de elementos de Fp

$$((0, w_0), (1, w_1), (2, w_2), \dots, (n, w_n))$$

 $((\sigma(0), w_0), (\sigma(1), w_1), (\sigma(2), w_2), \dots, (\sigma(n), w_n))$

Problema: Los vectores no son vectores de elementos de Fp

$$((0, w_0), (1, w_1), (2, w_2), \dots, (n, w_n))$$

 $((\sigma(0), w_0), (\sigma(1), w_1), (\sigma(2), w_2), \dots, (\sigma(n), w_n))$

Solución: "aplanarlos" artificialmente. Con β random:

$$(\beta\omega^{0} + w_{0}, \beta\omega^{1} + w_{1}, \beta\omega^{2} + w_{2}, \dots, \beta\omega^{n-1} + w_{n-1})$$
$$(\beta\omega^{\sigma(0)} + w_{0}, \beta\omega^{\sigma(1)} + w_{1}, \beta\omega^{\sigma(2)} + w_{2}, \dots, \beta\omega^{\sigma(n-1)} + w_{n-1})$$

Notación:

Para no escribir todo esto:

$$(\beta\omega^{0} + w_{0}, \beta\omega^{1} + w_{1}, \beta\omega^{2} + w_{2}, \dots, \beta\omega^{n-1} + w_{n-1})$$
$$(\beta\omega^{\sigma(0)} + w_{0}, \beta\omega^{\sigma(1)} + w_{1}, \beta\omega^{\sigma(2)} + w_{2}, \dots, \beta\omega^{\sigma(n-1)} + w_{n-1})$$

Llamemos a estos vectores:

$$V_1 = \beta D + W$$

$$V_2 = \beta \sigma(D) + W$$

En general

El vector (aplanado) W respeta la máscara M con alta probabilidad si el vector

$$V_1 = \beta D + W$$

es una permutación del vector

$$V_2 = \beta \sigma(D) + W$$

Prover

Verifier

Interpola W sobre D Obtiene *w*

Envía un oráculo [w]

Construye
$$V_1 = \beta D + W$$
 $V_2 = \beta \sigma(D) + W$

Construye Z usando a

Calcula

$$t = \frac{Z(X)(v_1 + \alpha) - Z(\omega X)(v_2 + \alpha)}{X^N - 1}$$

Envía oráculos [Z] y [t]

Recibe [w]

Sortea coeficientes α y β

Envía α y β al Prover

Recibe [Z] y [t]. Elige ζ random.

Calcula $a:=Z(\zeta)$ $b:=v_1(\zeta)+\alpha$ $c:=Z(\zeta\omega)$ $d:=v_2(\zeta)+\alpha$

 $e := t(\zeta) \cdot (\zeta^N - 1)$

Verifica ab -cd = e Verifica Z(1) != 0

A codear!