| **CS 203.4860** | **Secure Multi-Party Computation** | Fall 2021 |
|---|---|---|
| | Homework 1: Report | |
| *Lecturer: Dr. Adi Akavia* | | *Student(s): (Aya)* |

## problem 1

Truth table:

| $\alpha$ \x | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 1 | 1 |
| 3 | 0 | 0 | 1 | 1 |

DNF of truth table for equation (2) when every variable is presented as binary value:

$$(\neg a_1 \wedge a_0 \wedge \neg x_1 \wedge x_0) \cup (a_1 \wedge a_0 \wedge \neg x_1 \wedge x_0) \cup (\neg a_1 \wedge a_0 \wedge x_1 \wedge x_0) \cup (a_1 \wedge a_0 \wedge x_1 \wedge x_0) =$$

$$((\neg a_0 \cup a_0) \wedge a_1 \wedge \neg x_0 \wedge x_1) \cup ((\neg a_0 \cup a_0) \wedge a_1 \wedge x_0 \wedge x_1) =$$

$$(a_1 \wedge \neg x_0 \wedge x_1) \cup (a_1 \wedge x_0 \wedge x_1) = (a_1 \wedge (\neg x_0 \cup x_0) \wedge x_1) = (a_1 \wedge x_1)$$

## problem 2

$$\vec{a} = (a_1, a_2)$$

$$\vec{x} = (x_1, x_2)$$

$$\vec{a}, \vec{x} \in 0, 1, 2, 3^2$$

where each entry is specified in binary representation. Namely:

$$a_1 = (a_{11}, a_{10}) \in 0, 1^2, a_2 = (a_{21}, a_{20}) \in 0, 1^2, x_1 = (x_{11}, x_{10}) \in 0, 1^2, x_2 = (x_{21}, x_{20}) \in 0, 1^2$$

We will divide into cases and add OR gate between them:

Case 1:

$$a_1 x_1 \geq 4 \cup a_2 x_2 \geq 4 \tag{1}$$

and we'll use the DNF we found before for equation (2):

$$(a_{11} \wedge x_{11} \cup a_{21} \wedge x_{21})$$

We use the fact that:

$$A \cup B = (A \oplus B) \oplus (A \wedge B)$$

Then:

$$((a_{11} \wedge x_{11}) \cup (a_{21} \wedge x_{21})) =$$

| $\alpha$ \x | 00 | 01 | 10 | 11 |
|---|---|---|---|---|
| 00 | 0 | 0 | 0 | 0 |
| 01 | 0 | 0 | 0 | 0 |
| 10 | 0 | 0 | 1 | 1 |
| 11 | 0 | 0 | 1 | 1 |

$$((a_{11} \wedge x_{11}) \oplus (a_{21} \wedge x_{21})) \oplus ((a_{11} \wedge x_{11}) \wedge (a_{21} \wedge x_{21})) =$$

$$((a_{11} \wedge x_{11}) \oplus (a_{21} \wedge x_{21})) \oplus (a_{11} \wedge x_{11} \wedge a_{21} \cup x_{21})$$

<u>Case 2:</u>

$$a_1x_1 \geq 2 \cap a_2x_2 \geq 2 \tag{2}$$

| $\alpha$ \x | 00 | 01 | 10 | 11 |
|---|---|---|---|---|
| 00 | 0 | 0 | 0 | 0 |
| 01 | 0 | 0 | 1 | 1 |
| 10 | 0 | 1 | 1 | 1 |
| 11 | 0 | 1 | 1 | 1 |

for Implement that case we've used DNF and got that:

$$((a_{10} \cap x_{11}) \cup (a_{11} \cap x_{10})) \cap ((a_{20} \cap x_{21}) \cup (a_{21} \cap x_{20}))$$

We use the fact that:

$$A \cup B = (A \oplus B) \oplus (A \wedge B)$$

$$((a_{10} \cap x_{11}) \cup (a_{11} \cap x_{10})) \cap ((a_{20} \cap x_{21}) \cup (a_{21} \cap x_{20})) =$$

$((((a_{10} \cap x_{11}) \oplus (a_{11} \cap x_{10})) \oplus ((a_{10} \cap x_{11}) \wedge (a_{11} \cap x_{10}))) \cap ((((a_{20} \cap x_{21}) \oplus (a_{21} \cap x_{20})) \oplus ((a_{20} \cap x_{21}) \wedge (a_{21} \cap x_{20})))$

<u>Case 3:</u>

$$(a_1x_1 \geq 3 \cap a_2x_2 \geq 1) \tag{3}$$

Truth table for :$ax \geq 3$

| $\alpha$ \x | 00 | 01 | 10 | 11 |
|---|---|---|---|---|
| 00 | 0 | 0 | 0 | 0 |
| 01 | 0 | 0 | 1 | 0 |
| 10 | 0 | 1 | 1 | 1 |
| 11 | 0 | 0 | 1 | 1 |

Truth table for :$ax \geq 1$

| $\alpha$ \x | 00 | 01 | 10 | 11 |
|---|---|---|---|---|
| 00 | 0 | 0 | 0 | 0 |
| 01 | 0 | 1 | 1 | 1 |
| 10 | 0 | 1 | 1 | 1 |
| 11 | 0 | 1 | 1 | 1 |

for Implement that case we've used DNF and got that for $(a_1x_1 \geq 3 \cap a_2x_2 \geq 1)$

$((a_{11} \cap x_{11}) \cup (x_{11} \cap a_{10} \cap x_{10}) \cup (a_{10} \cap a_{11} \cap x_{10})) \cap ((x_{21} \cap a_{20}) \cup (x_{20} \cap a_{20}) \cup (x_{21} \cap a_{21})) \cup (x_{20} \cap a_{21})$

Case 4:

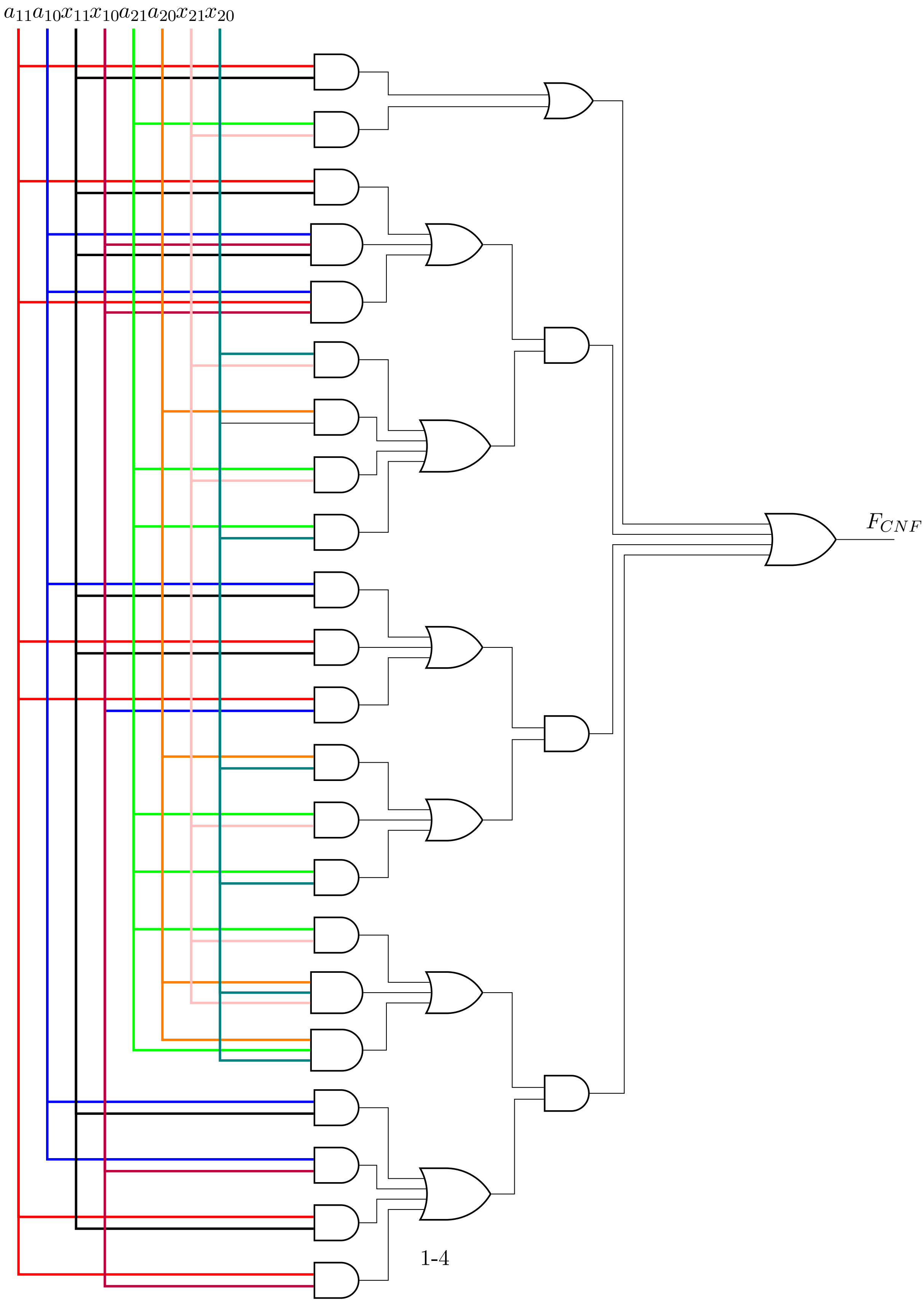
$$(a_1x_1 \geq 1 \cap a_2x_2 \geq 3) \tag{4}$$

for Implement that case we've used DNF and got that for $(a_1x_1 \geq 1 \cap a_2x_2 \geq 3)$

$((a_{21} \cap x_{21}) \cup (x_{21} \cap a_{20} \cap x_{20}) \cup (a_{20} \cap a_{21} \cap x_{20})) \cap ((x_{11} \cap a_{10}) \cup (x_{10} \cap a_{10}) \cup (x_{11} \cap a_{11})) \cup (x_{10} \cap a_{11})$

We do the same in case 3 to replace the OR and put XOR instead and then we put OR between case 1 case 2 case 3 case 4 and do the same to get XOR instead of OR.

**The boolean circuit:**



# problem 3

We first create a help circuit $G_4 : GF(11) \to GF(11)$ such that $G_4(a) = 0 \Leftrightarrow 0 \leq a < 4$ this can be easily done with

$$G(x) = (x \cdot (x - 1)) \cdot ((x - 2) \cdot (x - 3))$$

. we will use the following fact, for every two numbers a,b such $a + b \geq 4$ then $G_4(a) \vee G_4(b) \vee G_4(a + bmod11) = True$ (when considering 0 as false and true otherwise). We also notate the operation $B(a) = a^{10}$ (computed with 4 multiplications) Now we can easily construct a circuit

$$c_0 = \alpha_0 \cdot x_0, c_1 = \alpha_1 \cdot x_1$$

$$B((B(G(c_0)) + B(G(c_1))) + B(G(c_0 + c_1)))$$

when applying B on the sum acts as applying or on the outputs.

# problem 4

## 0.1 analysing problem 2

We can analyze circuit depth acording to the boolean circuit we draw before every OR gate is equal to 3 since $A \cup B = (A \oplus B) \oplus (A \wedge B)$ so according to our boolean circuit circuit depth=8, So on in calculating the size every OR gate is equal to 3,so circuit $size = 49$, MULT is equal to number of multiplication operations we can look in the circuit we draw and every OR is equal to one multiplication operation since we can change it to XOR by using this $A \cup B = (A \oplus B) \oplus (A \wedge B)$ so in this case $MULT = 33$,and the multiplicative depth (x-depth) that captures the maximal distance between an input and an output when counting only multiplication, considering OR as 1 multiplication x-depth=4

## 0.2 analysing problem 3

The depth is $depth(B)+2+depth(B)+depth(G)+1$, the or operation taking $depth(B)+2+depth(B)$ while the separate $G$ testing takes another $depth(G)$ plus one for the multiplication in the start. It is important to note that the addition of $c_0 + c_1$ adding one depth but later in the or operation this path takes one less. Therefore we get $depth = 2 \cdot 4 + 3 + 3 = 14$ The size of the circuit is 4 time B functions two plus operation for the or, 2 multiplications for $c_0, c_1$ and one addition for the input $c_0 + c_1$ and three g functions. Therefore we get $4 \cdot size(B) + 3 \cdot size(G) + 5 = 4 \cdot 4 + 3 \cdot 6 + 5 = 39$

the depth in term of multiplications is $depth_\times(B) + depth_\times(B) + depth_\times(G) + 1 = 4 + 4 + 2 + 1 = 11$

And the size in term of multiplication is $4 \cdot size_\times(B) + 3 \cdot size_\times(G) + 2 = 4 \cdot 4 + 3 \cdot 3 + 2 = 27$

# problem 6

We will first create a circuit $c_1$, evaluating function $sum^4 : \{0,1\}^3 \times \{0,1\}^3 \to \{0,1\}^3$ (considering the inputs as two numbers in the range $(0,7)$) with the following properties

$$x + y \geq 4 \Rightarrow sum^4(x, y) \geq 4$$

$$x + y < 4 \Rightarrow sum^4(x, y) = x + y$$

We will right formulas for the output of the circuit

$$O_0 = x_0 + y_0$$

$$O_1 = (x_1 + y_1) + (x_0 \cdot y_0)$$

$$O_2 = 1 + ((1 + x_1 \cdot y_1)((1 + x_2)(1 + x_1 \cdot (x_0 \cdot y_0)))((1 + y_2)(1 + y_1 \cdot (x_0 \cdot y_0))))$$

It easy to see that the composition of $sum^4$ is $\geq 4$ iff the sum is $\geq 4$, therefore we can use it to sum numbers. Additionally multiplication of $x = (x_0, x_1), y = (y_0, y_1)$ can be represented as $(x_0 \cdot y_0, x_0 \cdot y_1, 0) + (0, x_1 \cdot y_0, x_1 \cdot y_1)$, when for summation we can use $sum^4$

now we can directly compute $x_0 \cdot \alpha_0 + x_1 \cdot \alpha_1$ via $sum^4$ as showed above, and then output the msb of $x_0 \cdot \alpha_0 + x_1 \cdot \alpha_1$ indicating if it is grater or equal to 4.

## Analysing:

Every sum of two numbers take 9 bit-additions and 8 bit-multiplications, and multiplying two numbers take 4 bit-multiplications and 1 number summation of numbers. We have 2 multiplications and one summation equivalent to 4 bit-multiplications and 3 summations and that is equal to 28 bit-multiplications and 27 bit-additions.

the depth of the circuit is characterized by computing $O_2$ as this output take at least the same depth as the other gates, therefore we get the depth is 2 times the depth of computing $O_2$ (one times for summation and one time for the summation inside of the multiplication) and one time bit-multiplication of $x_{ij} \cdot \alpha_{ik}$, thus we get depth of 1 (for the first bit-multiplication) plus 8 for the first summation (the one inside the multipliction) and 6 for the second summation (in the second summation the input variables $x_2$ and $y_2$ have much bigger depth and therefore dominate the computation), this is depth of 15. The multiplicative depth is 5 for the first summation and 3 for the second one, and one for the bit-multiplication in the start of the multiplication resulting in depth of 9.