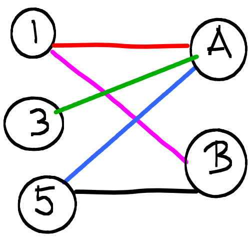
Logic

We need functions: next-state and output.
We can use {0,1} for both states and symbols ===> Boolean functions.

SET == Collection of Objects **E.g. S1 = { A, B }** **S2 = { 1, 3, 5 }**

RELATION (binary) == set of pairs

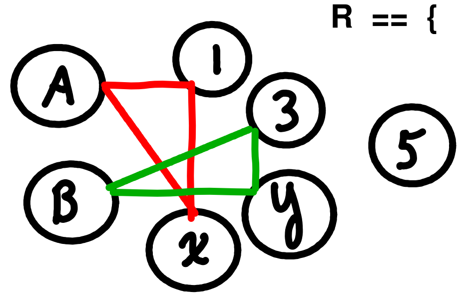


R = {
{1, A}, {3, A}, {5, A},
{1, B}, {5, B}
}

RELATION (ternary) == set of triples **E.g. S3 = { x, y }**

R == { {A, 1, x}, {B, 3, y} }

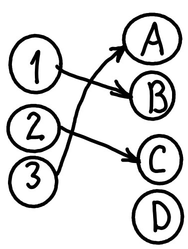
RELATION (k-ary) == set of k-tuples
elements are k-gons
with vertices from k sets



---

function

f : S1 ===> S2 **f "maps" elements of S1 to elements of S2**

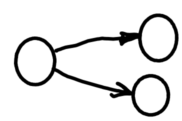


S1 = { 1, 2, 3 }
S2 = { A, B, C, D }

f = { (1,B), (2,C), (3,A) }

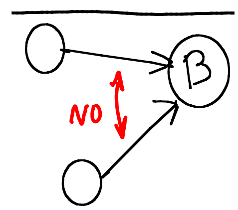
f(1) = B, f(2) = C, f(3) = A

a function because
this never happens

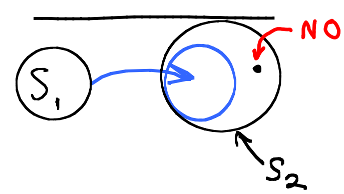


f(1) = B, f(1) = C

1-to-1



ON-TO



S1 X S2, "Set Cross Product"
Set of All Possible Pairs

$S_1 \backslash S_2$	A	B	C	D
1	(1,A)	(1,B)	(1,C)	(1,D)
2	(2,A)	(2,B)	(2,C)	(2,D)
3	(3,A)	(3,B)	(3,C)	(3,D)

$S_1 \times S_2$

k-ary operator

$$f : S^k \to S$$

k-way cross product, all possible k-tuples

Boolean variable x in { 0, 1 }   0 = "FALSE"   1 = "TRUE"

Boolean function $f: \{0,1\}^n \to \{0,1\}$ f maps Boolean n-tuples to {0, 1}

$\{0,1\}^2 \xrightarrow{f} \{0,1\}$

0 0	→	1
0 1	→	0
1 0	→	1
1 1	→	0

variable   function

Value of Variable   value of function

x	y	f(x, y)
0	0	1
0	1	0
1	0	1
1	1	0

What are the simplest functions we can imagine?

Can we build arbitrary functions?

Unary boolean functions: How many are there?

x		x		x		x	
0	$\boxed{0}$	0	0	0	1	0	1
1	$\boxed{0}$	1	1	1	0	1	1

Are any of these functions interesting? boring?

2-row output vector, 4 possibilities

$f(x) = 0$ $f(x) = x$ $f(x) = NOT(x)$ $f(x) = 1$

How many binary Boolean functions are there?   Which have names?

x y		x y		x y		x y		x y		x y		x y		x y	
0 0	0	0 0	0	0 0	0	0 0	0	0 0		0 0		0 0		0 0	
0 1	0	0 1	0	0 1	0	0 1	0	0 1		0 1		0 1		0 1	
1 0	0	1 0	0	1 0	1	1 0	1	1 0		1 0		1 0		1 0	
1 1	0	1 1	1	1 1	0	1 1	1	1 1		1 1		1 1		1 1	

x y		x y		x y		x y		x y		x y		x y		x y	
0 0		0 0		0 0		0 0		0 0		0 0		0 0	1	0 0	1
0 1		0 1		0 1		0 1		0 1		0 1		0 1	1	0 1	1
1 0		1 0		1 0		1 0		1 0		1 0		1 0	1	1 0	1
1 1		1 1		1 1		1 1		1 1		1 1		1 1	0	1 1	1

ALL Possible output vectors:

0000
0001
0010
...
1110
1111

2^4 = 16

How many k-ary functions are there?

			f
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

How many ways to form this column?

[0,0,0,0,0,0,0,0] ← 0

[0,0,0,0,0,0,0,1] ← 1

[0,0,0,0,0,0,1,0] ← 2

... ⋮

[1,1,1,1,1,1,1,0]

[1,1,1,1,1,1,1,1] ← $2^8 - 1$

Counting in binary
0 to (2^8 - 1)

== 2^8 functions

== 2^(2^3)

How many k-ary functions?

2^(2^k)

Back to our task:

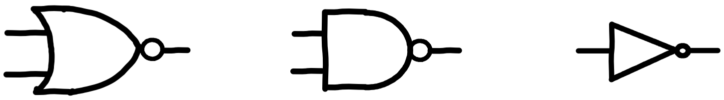
Build any arbitrary boolean function. Why?

Build a computer (UTM) ===> Build FSM

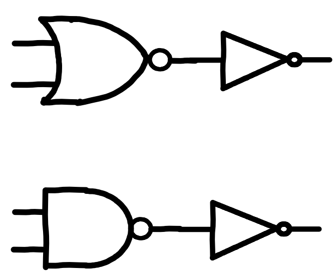
We need

--- arbitrary **next-state functions**
--- arbitrary **output functions**

what we can build, so far



NOR **NAND** **NOT**



OR

AND

Can we build EVERY

(1) **k-input**, **n-output** boolean function? (Seems hard.)

(2) **2-input**, **boolean** function? (Try something easier?)

(3) **2-input** function that outputs **exaclty one 1**?

The last one seems easiest.
Maybe we should explore Boolean functions a bit more first.

Propositions and Logical Connectives

(X **OR** Y) == TRUE
exactly when

(NOT(X) AND Y) == TRUE

(X AND NOT(Y)) == TRUE

(X AND Y) == TRUE

x	y	OR
0	0	0
0	1	1
1	0	1
1	1	1

A row in table identifies a state of the universe. f is either True or False when that is the state.

X = "**It is raining**"   **Y** = "**My hat is lost**"

(X + Y) is **TRUE** exactly when

(("**It is raining**" is FALSE) **AND** ("**My hat is lost**" is TRUE))

(("**It is raining**" is TRUE) **AND** ("**My hat is lost**" is FALSE))

(("**It is raining**" is TRUE) **AND** "(**My hat is lost**" is TRUE))

x	y	AND
0	0	0
0	1	0
1	0	0
1	1	1

(X AND Y) == TRUE
exactly when

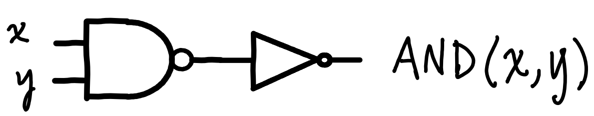
("**It is raining**" is TRUE) **AND** ("**My hat is lost**" is TRUE)

This function is simple: only one 1 in output

There is only 1 row, 1 state of the universe such that

$$AND(x,y) = 1 \longrightarrow (x = 1 \text{ and } y = 1)$$

We can build this function:



Are there other special functions like AND?

Can we build them easily?

Can we use them to build other, more complex functions?

Function Composition

chained mapping $S_1 \xrightarrow{g} S_2 \xrightarrow{h} S_3$

is some map $S_1 \xrightarrow{f} S_3$

$$f = h \circ g$$

$$f(x) = h(g(x))$$

1. Use composition to build a function from other functions.

E.g.,

X	NOT(X)
0	1
1	0

X	*f* (X) == NOT(NOT(X))
0	0
1	1

2. Can we find a set of functions we can use to build any other function?

---

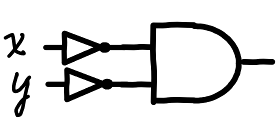
4 special binary functions

x	y	m_0
0	0	1
0	1	0
1	0	0
1	1	0

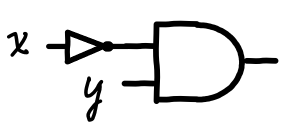
x	y	m_1
0	0	0
0	1	1
1	0	0
1	1	0

x	y	m_2
0	0	0
0	1	0
1	0	1
1	1	0

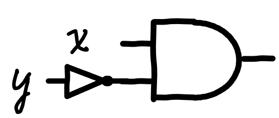
x	y	m_3
0	0	0
0	1	0
1	0	0
1	1	1



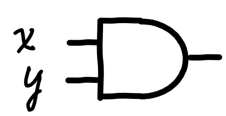
$\bar{x} \cdot \bar{y}$



$\bar{x} \cdot y$



$x \cdot \bar{y}$



$x \cdot y$

Binary Minterms

1. These 4 functions are orthogonal: They do not share any rows with a 1 output.

2. They are complete: Between them they cover all possible rows with a 1 output.

Can we combine them to form any other binary function?

Harmonic Analysis

Can we Compose simple functions.

Can we Decompose to simple functions?

x	y	f
0	0	0
0	1	1
1	0	0
1	1	1

$$= \begin{matrix}0\\1\\0\\0\end{matrix} \;\textbf{OR}\; \begin{matrix}0\\0\\0\\1\end{matrix} =$$

x	y	m_1
0	0	0
0	1	1
1	0	0
1	1	0

OR

x	y	m_3
0	0	0
0	1	0
1	0	0
1	1	1

$$f = m_1 + m_3$$

f is TRUE exactly when

m_1 is TRUE OR m_3 is TRUE

m_1 and m_3 are never 1 at the same time, they are orthogonal.

The set { m_0 , m_1 , m_2 , m_3 } is a complete set of orthogonal functions, binary minterms.

ANY binary function can be expressed as a sum of binary minterms.

We now can build any binary (2-bit input) function.

k-ary functions

Does this extend to k-bit input functions? YES.

X4	X3	X2	X1	X0
0	1	0	1	1

a row of truth table input values

is a logical statement

$$\overline{x}_4 \cdot x_3 \cdot \overline{x}_2 \cdot x_1 \cdot x_0$$

as a function it is true in exactly one case → a minterm function

What sort of functions are these minterms?

Can we express M_0? **M_0 is TRUE** **if and only if** **(A is FALSE) AND (B is FALSE) AND (C is FALSE)**

A	B	C	M_0
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

$$M_0 = \bar{A}\,\bar{B}\,\bar{C}\ ?$$

A	B	C	$\bar{A}\,\bar{B}\,\bar{C}$
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

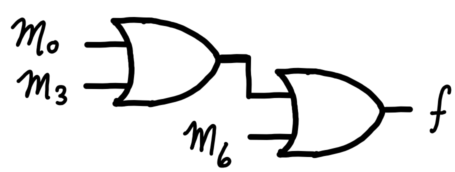
Yes, that is an equivalent expression for M0.

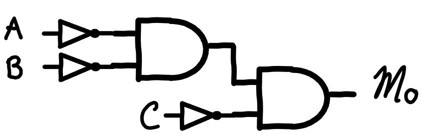
check that

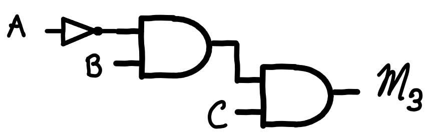
$$M_3 = \bar{A}\cdot B\cdot C$$
$$M_6 = A\cdot B\cdot \bar{C}$$

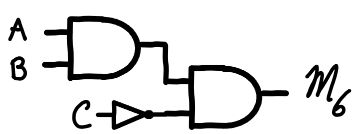
So, $$f = \bar{A}\,\bar{B}\,\bar{C} + \bar{A}\cdot B\cdot C + A\cdot B\cdot \bar{C}$$

Can we **build** f? YES, it consists of an OR of minterms.
Can we **build minterms**? YES, each consists of ANDs and NOTs.







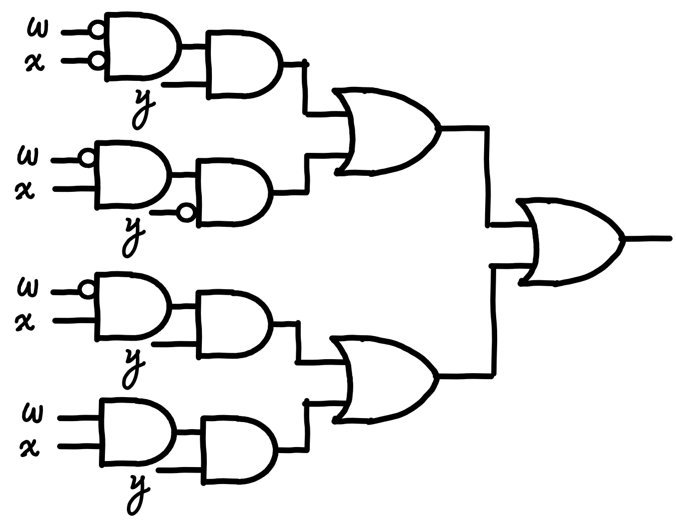


In general, **We can build ANY k-input function**:

A tree of NOTs, ANDs, and ORs.

$$f = \sum_{i=0}^{2^k - 1} a_i m_i \qquad a_i = 1 \text{ if } f(i) = 1, \quad a_i = 0 \text{ if } f(i) = 0$$

w	x	y	f	
0	0	0	0	
0	0	1	1	→ $\bar{w}\cdot\bar{x}\cdot y$
0	1	0	0	
0	1	1	1	→ $\bar{w}\cdot x\cdot\bar{y}$
1	0	0	1	→ $\bar{w}\cdot x\cdot y$
1	0	1	0	
1	1	0	0	
1	1	1	1	→ $w\cdot x\cdot y$



An AND-OR Tree.

We can build ANY Boolean function.

What if k-bit output?

Each output bit is a boolean function.

$\boxed{\text{what about symbols?}}$

$S = \{a, b, c, d\} \rightarrow \{00, 01, 10, 11\}$

Encode symbols as bit strings

$f : S \rightarrow S$

What about functions? **Maps from symbols to symbols?**

Each bit of output string is a boolean function.

		f
0	0	0 0
0	1	1 1
1	0	0 1
1	1	1 1

$f(a) = a$
$f(b) = d$
$f(c) = b$
$f(d) = d$

→

		f_1
0	0	0
0	1	1
1	0	0
1	1	1

		f_0
0	0	0
0	1	1
1	0	1
1	1	1