Boolean Logic

CS Fundamentals

What will we learn?

- What is boolean algebra/logic?
- Boolean relations and operators
- Bitwise logical operations
- Logic statements

What is boolean algebra (or boolean logic)?

A branch of mathematics that involves boolean values and operations on boolean values. It is used to break down complex problems, and is the heart of computer science.

In boolean logic, boolean values may be expressed as TRUE and FALSE, or 1 and 0.

Boolean operations

NOT, AND, OR, XOR

NOT relation and operators

NOT is a unary boolean relation (meaning it is evaluated on one expression instead of two)

NOT gives the opposite of the value on which it is applied (NOT true = false, NOT false = true)

Symbols that denote NOT:

¬ ~ !	negation not propositional logic	The statement $\neg A$ is true if and only if A is false. A slash placed through another operator is the same as ¬ placed in front.

Source of symbols: https://en.wikipedia.org/wiki/List_of_logic_symbols

AND relation and operators

Both values must be true for AND to evaluate to true

p	q	p∧q
T	T	T
T	F	F
F	T	F
F	F	F

∧ · &	logical conjunction and propositional logic, Boolean algebra	The statement *A* ∧ *B* is true if *A* and *B* are both true; otherwise, it is false.

OR relation and operators

One or the other of the values must be true for OR to evaluate to true (at least one)

p	q	p∨q
T	T	T
T	F	T
F	T	T
F	F	F

∨ + ∥	logical (inclusive) disjunction or propositional logic, Boolean algebra	The statement *A* ∨ *B* is true if *A* or *B* (or both) are true; if both are false, the statement is false.

XOR relation and operators

One or the other of the values **but not both** must be true for XOR to evaluate to true (the two values must be different)

p	q	$p \oplus q$
T	T	F
T	F	T
F	T	T
F	F	F

$\oplus$ $\veebar$	exclusive disjunction xor propositional logic, Boolean algebra	The statement $A \oplus B$ is true when either A or B, but not both, are true. $A \veebar B$ means the same.

Implication and Equivalence

Implication

Implication says that if p is true, then q is true. It is only false when p is true and q is false. Written as: $p \Rightarrow q$

If p is false, q can be either true or false, and the implication is still true.

p	q	$p \Rightarrow q$
T	T	T
T	F	F
F	T	T
F	F	T

Example: I will bring an umbrella if it is raining.

(I may still bring my umbrella if it isn't raining, but if it is raining and I don't bring my umbrella, the implication is false.)

Implication can be really tricky

Consider this statement:

$a \land \neg b \Rightarrow c$

When is this statement false (what must the values of a, b, and c be)?

Break it down to solve it:

- When is implication false? When the left side is true and the right side is false
- When is AND true? When the left and right side are both true
- So what must 'a' be? TRUE (or 1)
- So what must 'b' be? FALSE (or 0)
- So what must 'c' be? FALSE (or 0)

Therefore, the statement is false when: a=1, b=0, c=0

Equivalence

Equivalence says that q is true if and only if p is true. Written as: $p \Leftrightarrow q$

Either both p and q are true or both are false for equivalence to evaluate to true.

p	q	$p \Leftrightarrow q$
T	T	T
T	F	F
F	T	F
F	F	T

Example: I will work if and only if it is a weekday.

(If it is a weekday, I will always work. If it is not a weekday, I will never work.)

Logic Statements

Logic statements

Syntax used to express complex logic statements with a combination of symbols and logical operators

- Symbols represent statements that evaluate to true
- Operators used include AND, OR, NOT, XOR (and any of the symbols that can represent those operators)
- Not an if/else construct - something more general

Logic statement example

Statements:

- I will go to the movies if I'm off work, Star Wars is showing, the theater is open, and they have choc tops.

- I will to to the park if it's sunny, I'm off work, and I am not sick

Symbols:

w: I'm working
m: Star Wars is showing at the theater
t: The theater is open
c: The theater has choc tops
r: It's raining
s: I'm sick

I will go to the movies or I will go to the park:

!w & ((m & t & c) || (!r & !s))

(!w & m & t & c) || (!w & !r & !s)

Bitwise Operations

Bitwise operations

We can apply logical operations to a series of bits (to a binary number)

1100110 AND 111010

```
   1100110
   0111010
   -------
    100010
```

1100110 OR 111010

```
   1100110
   0111010
   -------
   1111110
```

1100110 XOR 111010

```
   1100110
   0111010
   -------
   1011100
```

What about 38 XOR 23?

1. Convert to binary
2. XOR
3. Convert back to decimal

1. $38 = 100110_2$
   $23 = 10111_2$

2.
```
  100110
  010111
  ------
  110001
```

3. $110001_2 = 32+16+1 = 49_{10}$