1. Logic

Discrete Mathematics

Boolean Logic

A *statement* is a sentence that is either true or false (but not both).

Which of the following sentences are statements?

The grass is green.

Buffalo can fly.

Where is Loei?

Welcome to Big C!

Boolean Logic

True and false are boolean values.

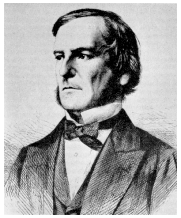
$\mathbb{B}$ = { false, true }

The grass is green — true

Buffalo can fly — false

Where is Loei? — not a statement

Welcome to Big C! — not a statement



George Boole (1815-1864) was an English mathematician and philosopher. His 'boolean system of logic' is used today in your computers, mobile phones, other electronic devices.

Boolean Logic

We can combine statements using AND, OR, NOT...

NOT (Buffalo can fly) — true

(The grass is green) AND (Thai food is delicious) — true

AND, OR, NOT are called 'boolean operators', because they are operations on boolean values.

(Same as +,-,*,/ are integer operators on integer values.)

Boolean operators

Five important operators:

$\neg A$	NOT A	Negation
$A \wedge B$	A AND B	Conjunction
$A \vee B$	A OR B	Disjunction
$A \Rightarrow B$	IF A THEN B	Implication
$A \Leftrightarrow B$	$A \Rightarrow B$ AND $B \Rightarrow A$	Equivalence

Boolean operators

Truth table for negation (NOT)

A	$\neg A$
FALSE	TRUE
TRUE	FALSE

Boolean operators

Truth table for conjunction (AND)

A	B	$A \wedge B$
FALSE	FALSE	FALSE
FALSE	TRUE	FALSE
TRUE	FALSE	FALSE
TRUE	TRUE	TRUE

Boolean operators

Truth table for disjunction (OR)

A	B	$A \vee B$
FALSE	FALSE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	TRUE

Boolean operators

Truth table for implication (IF THEN)

A	B	A ⇒ B
FALSE	FALSE	TRUE
FALSE	TRUE	TRUE
TRUE	FALSE	FALSE
TRUE	TRUE	TRUE

Boolean operators

Truth table for equivalence (IF AND ONLY IF)

A	B	A ⇔ B
FALSE	FALSE	TRUE
FALSE	TRUE	FALSE
TRUE	FALSE	FALSE
TRUE	TRUE	TRUE

Boolean operators

Similar to integer operators, we can write compound statements with multiple terms, and group using brackets, e.g. $(A \Rightarrow B) \wedge (B \Rightarrow C)$.

$(A \wedge B) \wedge C$ is the same as $A \wedge (B \wedge C)$,

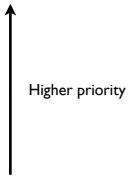
so it can written without brackets: $A \wedge B \wedge C$

However, $(A \wedge B) \vee C$ is not the same as $A \wedge (B \vee C)$.

Boolean operators

There is an 'order of precedence' (priority) for boolean operators:

- $\neg$ NOT
- $\wedge$ AND
- $\vee$ OR
- $\Rightarrow$ IF THEN
- $\Leftrightarrow$ IF AND ONLY IF



(generally the same as the order of precedence in programming languages like Java)

Hence $\neg A \wedge B \Rightarrow C$ means $(((\neg A) \wedge B) \Rightarrow C)$

Puzzle Break!

You have 8 coins, all the same size and weight, except for one which is heavier.

You only have two chances to use the weighing scales.

How can you find the heavier coin?

Truth tables

Truth tables completely define logical operators, and let us find the value of compound statements

Example: $(\neg A \Rightarrow B) \Leftrightarrow (\neg B \Rightarrow A)$

A	B	¬A	¬A⇒B	¬B	¬B⇒A	(¬A⇒B)⇔ (¬B⇒A)
F	F	T	F	T	F	T
F	T	T	T	F	T	T
T	F	F	T	T	T	T
T	T	F	T	F	T	T

$(\neg A \Rightarrow B) \Leftrightarrow (\neg B \Rightarrow A)$ is true for all A,B

Truth tables

Using truth tables is not always convenient.

Why?

Imagine trying to write the truth table for

$\neg((\neg A \wedge B) \vee (C \wedge \neg D)) \Leftrightarrow (A \wedge \neg C) \vee (A \wedge D) \vee (\neg B \wedge \neg C) \vee (\neg B \wedge D)$

To simplify Boolean expressions, we will use laws of logic.

Laws of logic

$\neg\neg A \rightarrow A$ (double negation law)

$A \wedge A \rightarrow A$ (idempotent law)
$A \vee A \rightarrow A$

$A \wedge B \rightarrow B \wedge A$ (commutative law)
$A \vee B \rightarrow B \vee A$

$(A \wedge B) \wedge C \rightarrow A \wedge (B \wedge C)$ (associative law)
$(A \vee B) \vee C \rightarrow A \vee (B \vee C)$

$A \wedge (B \vee C) \rightarrow (A \wedge B) \vee (A \wedge C)$ (distributive law)
$A \vee (B \wedge C) \rightarrow (A \vee B) \wedge (A \vee C)$

Laws of logic

$\neg (A \wedge B) \rightarrow \neg A \vee \neg B$ (De Morgan's laws)
$\neg (A \vee B) \rightarrow \neg A \wedge \neg B$

Thus,
$A \wedge B \rightarrow \neg(\neg A \vee \neg B)$,
so $\wedge$ can be expressed via $\neg, \vee$

Also,
$A \vee B \rightarrow \neg(\neg A \wedge \neg B)$,
so $\vee$ can be expressed via $\neg, \wedge$
(Cannot remove both $\wedge, \vee$ at the same time!)

Laws of logic

$A \wedge T \rightarrow A$ (identity laws)
$A \vee F \rightarrow A$
$A \wedge F \rightarrow F$ (annihilation laws)
$A \vee T \rightarrow T$
$A \wedge \neg A \rightarrow F$ (laws of excluded middle)
$A \vee \neg A \rightarrow T$
$A \wedge (A \vee B) \rightarrow A \rightarrow A \vee (A \wedge B)$ (absorption laws)

$(A \Rightarrow B) \rightarrow (\neg A \vee B) \rightarrow \neg (A \wedge \neg B)$ (implication law)

$(A \Leftrightarrow B) \rightarrow (A \Rightarrow B) \wedge (B \Rightarrow A) \rightarrow (A \wedge B) \vee (\neg A \wedge \neg B)$ (equivalence law)

So, $\Rightarrow$ and $\Leftrightarrow$ are redundant (but convenient)

Laws of logic

A statement that is always true is called a *tautology*.

A statement that is always false is called a *contradiction*.

Examples:

$A \vee \neg A$ tautology (because $A \vee \neg A \rightarrow T$)

$A \wedge \neg A$ contradiction (because $A \wedge \neg A \rightarrow F$)

Logical equivalences

Instead of using truth tables, we can use laws to show when two statements are equivalent.

Example:

Show that $\neg(p \vee (\neg p \wedge q))$ and $\neg p \wedge \neg q$ are equivalent.

Proof:

$\neg(p \vee (\neg p \wedge q)) \rightarrow \neg p \wedge \neg(\neg p \wedge q)$ (second De Morgan's law)
$\rightarrow \neg p \wedge (\neg(\neg p) \vee \neg q)$ (first De Morgan's law)
$\rightarrow \neg p \wedge (p \vee \neg q)$ (double negation law)
$\rightarrow (\neg p \wedge p) \vee (\neg p \wedge \neg q)$ (distributive law)
$\rightarrow F \vee (\neg p \wedge \neg q)$ (excluded middle law)
$\rightarrow \neg p \wedge \neg q$ (identity law)

Logical equivalences

Example:

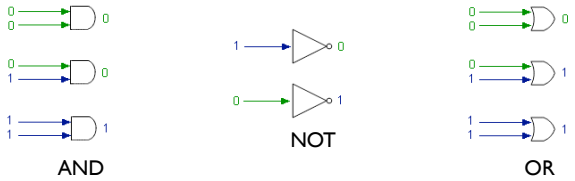
Show that $(p \wedge q) \Rightarrow (p \vee q)$ is a tautology.

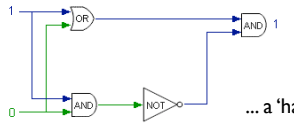
Proof:

$(p \wedge q) \Rightarrow (p \vee q)$	$\rightarrow \neg(p \wedge q) \vee (p \vee q)$	(implication law)
	$\rightarrow (\neg p \vee \neg q) \vee (p \vee q)$	(De Morgan's first law)
	$\rightarrow \neg p \vee p \vee \neg q \vee q$	(associative law)
	$\rightarrow T \vee \neg q \vee q$	(excluded middle law)
	$\rightarrow T$	(annihilation law)

Applications in hardware

In computer hardware, thousands of logic gates are joined together to allow the computer to execute machine code...





... a 'half-adder'

Applications in hardware

Chip makers would like to reduce the number of logic gates (because of size, cost, efficiency, heat, etc)...

So, we can use laws of logic to simplify boolean operations.

Applications in programming

Almost all programming languages have a boolean datatype, as well as the basic boolean operators. These are often used for comparisons...

```
bool b = (! x > 0) && y == 1 || z < 99;
```

How many boolean operators? Where are they?

Boolean: `!` `&&` `||`

Integer: `>` `==` `<` (but return boolean result)

Applications in search

Most search engines use boolean operators to specify search terms.

Examples (from Google Search):

"to be or not to be" matches the whole string

"to be" OR "not to be" matches the first string or the second string

"to be" "not to be" matches the first string and the second string

Summary

1. Statements are true or false.
2. Compound statements are constructed using boolean operators.
3. Use truth tables for showing all possibilities for a statement.
4. Compound statements can be simplified using laws of logic.

So far we have focussed on *Propositional* Logic (see sections 1.1 and 1.2 in "Discrete Mathematics and its applications" by Rosen).

Next we will look at *Predicate* Logic!