Boolean Algebra

Operations on truth values (true and false).

Fundamentally important for Switching Theory, Programming and Logic.
Let $B = \{0,\ 1\}$. 0 denotes FALSE, 1 denotes TRUE.

We define three operations (functions on B):
$\wedge : (B \times B) \to B$: AND.
$\vee : (B \times B) \to B$: OR.
$* : B \to B$: NOT.

Definition of $\wedge$ (AND):

$$\begin{aligned} 0 \wedge 0 &= 0 \\ 0 \wedge 1 &= 0 \\ 1 \wedge 0 &= 0 \\ 1 \wedge 1 &= 1 \end{aligned}$$

Thus: $x \wedge y = 1$ iff both x and y are 1.

Also written as $x \& y$ in logic, `(and x y)` in LISP.

Definition of $\vee$ (OR):

$$\begin{aligned} 0 \vee 0 &= 0 \\ 0 \vee 1 &= 1 \\ 1 \vee 0 &= 1 \\ 1 \vee 1 &= 1 \end{aligned}$$

Thus: $x \vee y = 1$ iff at least one of x and y is 1.

Also written as `(or x y)` i LISP.

Definition of $*$ (NOT):

$$\begin{aligned} 0^* &= 1 \\ 1^* &= 0 \end{aligned}$$

Thus: 1 becomes 0 and *vice versa*.

Also written as $\sim x$ or $\neg x$ in logic, $\overline{x}$ or x' in switching theory,
`(not x)` in LISP.

The Operations with List Notation

$$\begin{aligned} &\wedge : (B \times B) \to B \\ &\wedge = \{\langle\langle 0,0\rangle,0\rangle, \langle\langle 0,1\rangle,0\rangle, \langle\langle 1,0\rangle,0\rangle, \langle\langle 1,1\rangle,1\rangle\} \end{aligned} \tag{1}$$

$$\begin{aligned} &\vee : (B \times B) \to B \\ &\vee = \{\langle\langle 0,0\rangle,0\rangle, \langle\langle 0,1\rangle,1\rangle, \langle\langle 1,0\rangle,1\rangle, \langle\langle 1,1\rangle,1\rangle\} \end{aligned} \tag{2}$$

$$* : B \to B, \quad * = \{\langle 0,1\rangle, \langle 1,0\rangle\} \tag{3}$$

Exempel (Boolean Algebra):

$$\begin{aligned} &(0 \wedge (1^* \vee 1)) \vee (0 \wedge 1) = \\ &= (0 \wedge (0 \vee 1)) \vee (0 \wedge 1) \\ &= (0 \wedge 1) \vee (0 \wedge 1) \\ &= 0 \vee (0 \wedge 1) \\ &= 0 \vee 0 \\ &= 0 \end{aligned}$$

Properties of th Boolean Operations

1. Idempotent Laws:
   a) $(x \vee x) = x$, b) $(x \wedge x) = x$.
2. Commutative Laws:
   a) $(x \vee y) = (y \vee x)$, b) $(x \wedge y) = (y \wedge x)$.
3. Associative Laws:
   a) $(x \vee y) \vee z = z \vee (y \vee x)$
   b) $(x \wedge y) \wedge z = x \wedge (y \wedge z)$.
4. Distributive Laws:
   a) $x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$
   b) $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$.

5. Identity:
   a) $x \vee 0 = x$, b) $x \vee 1 = 1$,
   c) $x \wedge 0 = 0$, d) $x \wedge 1 = x$.

6. Complement Laws:
   a) $x \vee x^* = 1$, b) $(x^*)^* = x$, c) $x \wedge x^* = 0$.

7. DeMorgan's Laws:
   a) $(x \vee y)^* = x^* \wedge y^*$
   b) $(x \wedge y)^* = x^* \vee y^*$.

Hmm...

The laws for the boolean operations are strikingly similar to the laws of the set operations.

Let's compare!

Comparison: Boolean and Set Operations

Compare $\wedge$ (AND) and intersection $\cap$:

$$\begin{array}{ll} 0 \wedge 0 = 0 & \emptyset \cap \emptyset = \emptyset \\ 0 \wedge 1 = 0 & \emptyset \cap U = \emptyset \\ 1 \wedge 0 = 0 & U \cap \emptyset = \emptyset \\ 1 \wedge 1 = 1 & U \cap U = U \end{array}$$

Compare $\vee$ (OR) and union $\cup$:

$$\begin{array}{ll} 0 \vee 0 = 0 & \emptyset \cup \emptyset = \emptyset \\ 0 \vee 1 = 1 & \emptyset \cup U = U \\ 1 \vee 0 = 1 & U \cup \emptyset = U \\ 1 \vee 1 = 1 & U \cup U = U \end{array}$$

Compare $*$ (NOT) and complement $'$:

$$\begin{array}{ll} 0^* = 1 & \emptyset' = U \\ 1^* = 0 & U' = \emptyset \end{array}$$

Isomorphisms

There are obvious similarities between set algebra and boolean algebra.

- 0 corresponds to $\emptyset$ and 1 corresponds to U.
- $\vee$ corresponds to $\cup$ and $\wedge$ corresponds to $\cap$.
- $*$ corresponds to $'$.

We say that the two systems $\langle \{0,1\}, \vee, \wedge, *\rangle$ are $\langle \{\emptyset, U\}, \cup, \cap, '\rangle$ are *isomorphic*.

One more law...

Absorption

a) $x \vee (x \wedge y) = x$, b) $x \wedge (x \vee y) = x$.

A corresponding law in the set algebra:

a) $A \cup (A \cap B) = A$, b) $A \cap (A \cup B) = A$.

An example:

$(a \wedge b) \vee (b \wedge c) = b \wedge (c \vee a)$.

Another example:

$((x \vee y) \wedge y^*) \vee ((x \vee y) \wedge x^*) = (x^* \wedge y) \vee (x \wedge y^*)$.

Logical Problem Solving with Boolean Algebra: The Dinner Invitation Problem

i) A will only go if B goes, and
ii) B and C are enemies, and
iii) One of D and E has to be invited, and
iv) E cannot go anywhere without both A and D.

The Dinner Invitation Problem Cont'd

Study the first constraint:
i) A will only go if B goes.

a will mean "invite A", b will mean "invite B".

Either $a = 1$ (TRUE), which means that A is invited, or $a = 0$ (FALSE), which means that A is not invited. Note: $a = 0$ is equivalent to $a^* = 1$.
Same thing for b *etc.*

The Dinner Invitation Problem Cont'd

We have
i) A will only go if B goes.

$b \vee a^*$.
"Invite B or do not invite A", i.e. if we do not invite B, we cannot invite A either.

The Dinner Invitation Problem Cont'd

ii) B and C are enemies.

b means "invite B", and c means "invite C".

$(b \wedge c)^*$.
"Do not invite both B and C".

The Dinner Invitation Problem Cont'd

iii) One of D and E has to be invited.

$d \vee e$.

iv) E cannot go anywhere without both A and D.

$e^* \vee (a \wedge d)$.
"Do not invite E, or invite both A and D"

The Dinner Invitation Problem Cont'd

A formulation of the problem in boolean algebra.
i) $(b \vee a^*)\ \wedge$
ii) $(b \wedge c)^*\ \wedge$
iii) $(c \vee e)\ \wedge$
iv) $(e^* \vee (a \wedge d))$.

All four of these constraints have to be true, i.e. the expression
$(b \vee a^*)\ \wedge\ (b \wedge c)^*\ \wedge\ (d \vee e)\ \wedge\ (e^* \vee (a \wedge d))$ must be $= 1$ (TRUE)!

The Dinner Invitation Problem Cont'd

What solutions do we have to the equation
$(b \vee a^*)\ \wedge\ (b \wedge c)^*\ \wedge\ (d \vee e)\ \wedge\ (e^* \vee (a \wedge d))\ =\ 1$?

A solution is an *assignment* of a truth value to each of a, b, c, d, and e.

The Dinner Invitation Problem Cont'd

Let's start by trying to invite all of them, i.e.
$a = 1,\ b = 1,\ c = 1,\ d = 1,\ e = 1.$

$$(b \vee a^*)\ \wedge\ (b \wedge c)^*\ \wedge\ (d \vee e)\ \wedge\ (e^* \vee (a \wedge d)) \tag{4}$$
$$\wedge\ (1^* \vee (1 \wedge 1)) \tag{5}$$
$$(6)$$

The Dinner Invitation Problem Cont'd

Not a solution! But, what if we try to only invite A, B, and D? $a=1,\ b=1,\ c=0,\ d=1,\ e=0.$

$$(b\vee a^*)\ \wedge\ (b\wedge c)^*\ \wedge\ (d\vee e)\ \wedge\ (e^*\vee(a\wedge d))\ = \tag{7}$$

(8)

The Dinner Invitation Problem

$$\begin{aligned}&(b\vee a^*)\ \wedge\ (b\wedge c)^*\ \wedge\ (d\vee e)\ \wedge\ (e^*\vee(a\wedge d))\ =\\ &= (b\vee a^*)\ \wedge\ \underline{(b^*\vee c^*)}\ \wedge\ (d\vee e)\ \wedge\ \underline{((e^*\vee a)\wedge(e^*\vee d))}\\ &= (b\vee a^*)\ \wedge\ (b^*\vee c^*)\ \wedge\ (d\vee e)\ \wedge\ (e^*\vee a)\wedge(e^*\vee d)\end{aligned}$$

Conjunctive Normal Form (conjunction = "AND"). Fairly easy to spot solutions: A least one expression within every parenthesis must be 1. E.g. from the first parenthesis: $b=1$ or $a^*=1$ (which means that $a=0$). There is also a Disjunctive Normal Form.

Switching Theory

Logic (propositional logic)

We usually write T and F or $\top$ and $\bot$ instead of 1 and 0. Moreover, we use $\neg a$ or $\sim a$ instead of a^*.
Example: $c\wedge\neg(a\vee b)$

Logic Cont'd

In propositional logic a, b and c (or p, q, r) are seen as statements (propositions) that either are true or false, and $\neg$, $\wedge$ and $\vee$ are used to construct comlex (composite) statements.
E.g. if $p=$ "it rains" and $q=$ "the ground gets wet", we can interpret $p\wedge q$ as "it rains and the ground gets wet".

Logic Cont'd

We often use the operation $\rightarrow$ (IF ... THEN ...) and $\leftrightarrow$ (... IFF ...).

$$(p\rightarrow q)\quad =_{def}\quad (\neg p\vee q)$$
$$(p\leftrightarrow q)\quad =_{def}\quad ((p\rightarrow q)\wedge(q\rightarrow p))$$

E.g. if $p=$ "it rains" and $q=$ "the ground gets wet", we can interpret $p\rightarrow q$ as "if it rains, the ground gets wet".

Summary

- Two values: 1 for "TRUE" and 0 for "FALSE"
- Three operations: $\wedge$ for "AND", $\vee$ for "OR" and * for "NOT".
- Very similar to set algebra.
- Useful in: Programming, Switching Theory, **Logic**.