Notes by Ioan Steffan Thomas

Boolean Algebra

Invented by George Boole (1815-1864), in his book "The Laws of Thought" in 1854.

Further reading: https://en.wikipedia.org/wiki/Boolean_algebra_(structure)

A Boolean Algebra is a tuple:

$$(B, 0,1, +, -)$$

- B is a nonempty set, B represents the domain of discourse.
- $0,1 \in B$.
- $+: B \times B \to B$, a binary function on B.
- $-: B \to B$, a unary function on B.

Variables in Boolean algebra are often written in lowercase.

Syntactic abbreviations:

$$-a \equiv \overline{a}$$
$$a \cdot b \equiv -\left(\overline{a} + \overline{b}\right)$$

The bullet operator ($\cdot$) is called a product, the plus operator ($+$) is a sum, and the minus operator ($-$) is a negation or complement.

Precedence of operators, from highest to lowest: $-$, $\cdot$, $+$.

Axioms of Boolean Algebra:

- Axioms for $+$:
  - $a + (b + c) \equiv (a + b) + c$ (associativity)
  - $a + b \equiv b + a$ (commutativity)
  - $a + a \equiv a$ (idempotency)
  - $a + 0 \equiv a$ (unit/identity law)
- Axioms for $-$:
  - $- - a \equiv a$ (double negation)
  - $a + \overline{a} \equiv 1$ (complement)
  - $-1 \equiv 0$ (0/1 law)
- Distribution Law:
  - $a \cdot (b + c) \equiv a \cdot b + a \cdot c$

Duality: the duality principle in Boolean algebra works in the same way as in propositional logic.

Consequences of the Axioms:

- De Morgan's laws: $-(a + b) = -a \cdot -b$, $-(a \cdot b) = -a + -b$, can be seen from definition of $\cdot$ in terms of – and $+$ and also the double negation law.

  $a \cdot \overline{a} \equiv 0$ (work from $\overline{a} + \overline{\overline{a}} \equiv 1$)
  $a \cdot 1 \equiv a$
  $a \cdot 0 \equiv 0$
  $a + 1 \equiv 1$ (work from)
  $1 \equiv -0$
  $a + (b \cdot c) \equiv (a + b) \cdot (a + c)$ (dual distribution law, by taking the dual of the prior distribution over negation variables)

Notes by Ioan Steffan Thomas

The traditional Boolean Algebra is given by the tuple:

$$(\{0,1\}, 0, 1, +, -)$$

- $B = \{0,1\}$
- $+: \begin{matrix} 0 & if\ 0+0 \\ 1 & otherwise \end{matrix}$
- $-: \begin{matrix} 0 \mapsto 1 \\ 1 \mapsto 0 \end{matrix}$
- This Boolean algebra can be shown equivalent to propositional logic, in the sense that propositional logic is the Boolean algebra $(\{\bot, \top\}, \bot, \top, \vee, \neg)$, and hence $\cdot$ is $\wedge$.

A Boolean Set Algebra is given by the tuple:

$$(\mathcal{P}(X), \emptyset, X, \cup, X\backslash)$$

- X is a nonempty set.
- $\mathcal{P}(X)$ is the powerset of X.
- $\emptyset$ is the empty set.
- $\cup$ is the union of two sets.
- $X\backslash$ is the complement with respect to X.
- $Theorem\ (Stone,\ 1936)$: every Boolean Algebra is isomorphic to a Boolean set algebra.

Sum of Products:

- Every Boolean term is equivalent to a sum of products. A sum of products is the same as DNF in propositional logic, i.e. in the form of product clauses (using $\cdot$) of literals (any Boolean variable or it's negation) being connected as a sum (using $+$).
- Sums of products can be constructed in the same way as DNF.
- Similarly, a product of sums is the same as CNF.

NAND:

- The Boolean operation NAND (negation of logical AND) can be defined as $a\ NAND\ b \equiv \overline{(a \cdot b)} \equiv \overline{a} + \overline{b}$, and has a truth table to match.
- $\overline{a} \equiv a\ NAND\ 1, a + b \equiv \overline{a}\ NAND\ \overline{b}$, so it is clear that negation can be represented by only NAND, and sum by only NAND and negation, hence since the axioms of Boolean algebra can be represented by only negation and sum operations, a NAND operation is the minimal required operation for representing all operations in a Boolean algebra.
- Note $A\ NAND\ (B\ NAND\ C) \not\equiv (A\ NAND\ B)\ NAND\ C$, i.e. NAND is not associative.

Boolean Algebra to as Logic Circuits:

- A logic circuit takes given input 'signals' (0 or 1), and outputs the result of performing a corresponding logical operation on the inputs.
- Boolean algebra can be used to model logic circuits.
- *Figure 1* show the more common logic 'gates', and their logical meaning. The inputs come from the left, and outputs on the right. The figure also uses $a.\boldsymbol{b}$ for $\boldsymbol{a} \cdot b$.

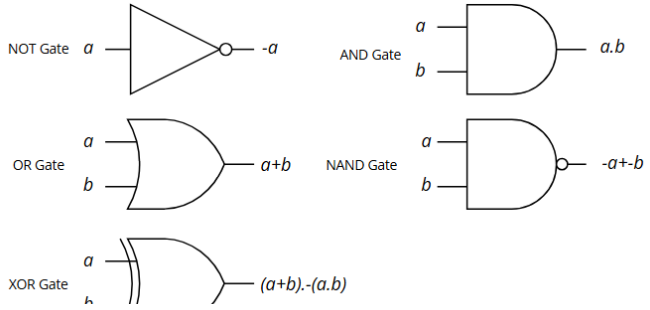

Figure 1: Common Logic Gates