

Logic

1. Boolean Logic
2. First Order Logic; Number Theory
3. Second Order Logic.

Connections between computation, complexity, and logic.

Boolean Logic: Syntax

Let $X = \{x_1, x_2, \dots\}$ be a set of Boolean variables.

We can combine Boolean variables by Boolean connectives \vee, \wedge, \neg .

Definition 1. A **Boolean expression** can be any one of

(a) *Boolean variable (such as x_i)*

(b) *an expression of the form $\neg\phi_1$, where ϕ_1 is a Boolean expression.*

(c) *$\phi_1 \vee \phi_2$, where ϕ_1 and ϕ_2 are boolean expressions.*

(d) *$\phi_1 \wedge \phi_2$, where ϕ_1 and ϕ_2 are boolean expressions.*

A **literal** is of the form x_i or $\neg x_i$ where x_i is a boolean variable.

Two other connectives: $\phi_1 \Rightarrow \phi_2$ is $\neg\phi_1 \vee \phi_2$ and $\phi_1 \Leftrightarrow \phi_2$ is $(\phi_1 \Rightarrow \phi_2) \wedge (\phi_2 \Rightarrow \phi_1)$.

Boolean Logic: Semantics

Definition 2. A truth assignment T is a mapping from a finite set X' of Boolean variables to $\{true, false\}$.

Given a boolean expression ϕ let $X(\phi)$ be the set of boolean variables **appearing** in ϕ .

Let T be a truth assignment that is **appropriate** to ϕ , i.e., it is defined on a set of Boolean variables such $X(\phi) \subset X'$.

We say that T **satisfies** ϕ ($T \models \phi$) if:

Case 1. ϕ is a variable x_i : $T(x_i) = true$.

Case 2. $\phi = \neg\phi_1$: $T \not\models \phi_1$.

Case 3. $\phi = (\phi_1 \vee \phi_2)$: $T \models \phi_1$ or $T \models \phi_2$.

Case 4. $\phi = (\phi_1 \wedge \phi_2)$: $T \models \phi_1$ and $T \models \phi_2$.

Example: Let $\phi = ((\neg x_1 \vee x_2) \wedge x_3)$ and let $T(x_1) = true$, $T(x_3) = true$ and $T(x_2) = false$. $T \not\models \phi$.

Boolean expression Properties

Definition 3. *Two expressions ϕ_1 and ϕ_2 are equivalent ($\phi_1 \equiv \phi_2$) if for any truth assignment T appropriate to both of them, $T \models \phi_1$ iff $T \models \phi_2$.*

Boolean connectives satisfy, commutativity, associativity, distributive, idempotent, and DeMorgan's laws.

Examples:

1. $(\phi_1 \vee \phi_2) \equiv (\phi_2 \vee \phi_1)$.
2. $((\phi_1 \vee \phi_2) \vee \phi_3) \equiv (\phi_1 \vee (\phi_2 \vee \phi_3))$.
3. $((\phi_1 \vee \phi_2) \wedge \phi_3) \equiv ((\phi_1 \wedge \phi_3) \vee ((\phi_2 \vee \phi_3)))$.
4. $\phi \vee \phi \equiv \phi$.
5. $\neg(\phi_1 \wedge \phi_2) \equiv (\neg\phi_1 \vee \neg\phi_2)$.

Can be proved by using "truth table method".

CNF and DNF Forms

A boolean expression is in **Conjunctive normal form (CNF)** if $\phi = \bigwedge_{i=1}^n C_i$ and each of the C_i 's is the disjunction of one or more literals (called **clauses**).

A boolean expression is in **Disjunctive normal form (DNF)** if $\phi = \bigvee_{i=1}^n D_i$ and each of the D_i 's is the conjunction of one or more literals (called **implicants**).

Theorem 1. *Every Boolean expression is equivalent to one in CNF and to one in DNF.*

Proof

By induction on the structure of ϕ :

1. ϕ is a single variable: trivial.

2. $\phi = \neg\phi_1$: By induction, ϕ_1 is equivalent to some DNF expression. Apply DeMorgan's law to convert to CNF. Similar argument for DNF.

3. $\phi = (\phi_1 \vee \phi_2)$: Trivial for DNF.

For CNF, by induction, ϕ_1 and ϕ_2 are in CNF.

Let $\{C_{1i} : i = 1, \dots, n_1\}$ and $\{C_{2j} : j = 1, \dots, n_2\}$ be the two sets of clauses.

Then $\{(C_{1i} \vee C_{2j}) : i = 1, \dots, n_1, j = 1, \dots, n_2\}$ is equivalent to ϕ .

4. $\phi = (\phi_1 \wedge \phi_2)$: Symmetric to above.