Chapter 4. Boolean Logic

Note. In this short chapter, we define and illustrate Boolean variables, Boolean expressions, and Boolean functions. In each case, we are ultimately interested in assigning truth values of **true** or **false**.

4.1. Boolean Expressions

Note. In this section, we define Boolean variables which take on the truth values **true** and **false**. We then define elementary Boolean expressions (involving "not," "and," and "or") and use an inductive definition to define more complicated Boolean expressions. We state relations between Boolean expressions (in Proposition 4.1), define conjunctive and disjunctive normal forms, and prove that every Boolean expression is equivalent to one in conjunctive (and to one in disjunctive) normal form (in Theorem 4.1).

Definition. We consider a countably infinite alphabet of *Boolean variables* $X = \{x_1, x_2, \ldots\}$, where each variable can take on the *truth values* **true** and **false**. Boolean variables are combined using *Boolean connectives*, including $\vee$ (the *logical or*), $\wedge$ (the *logical and*), and $\neg$ (the *logical not*).

Note. By saying "logical" or/and/not, we can address the truth values of Boolean expressions (to be formally defined next) with truth tables, as you would in Mathematical Reasoning (MATH 3000); see my online notes for Mathematical Reasoning on Section 1.2. Logical Connectives and Truth Tables. We have the following tables:

x_1	x_2	$x_1 \vee x_2$
false	**false**	**false**
false	**true**	**true**
true	**false**	**true**
true	**true**	**true**

x_1	x_2	$x_1 \wedge x_2$
false	**false**	**false**
false	**true**	**false**
true	**false**	**false**
true	**true**	**true**

x	$\neg x$
false	**true**
true	**false**

Definition 4.1. A *Boolean expression* can be any one of (a) a Boolean variable, such as x_i, or (b) an expression of the form $\neg\phi_i$, where ϕ_1 is a Boolean expression, or (c) an expression of the form $(\phi_1 \vee \phi_2)$, where ϕ_1 and ϕ_2 are Boolean expressions, or (d) an expression of the form $(\phi_1 \wedge \phi_2)$ where ϕ_1 and ϕ_2 are Boolean expressions. Expression $\neg\phi_1$ is the *negation* of ϕ_1. Expression $(\phi_1 \vee \phi_2)$ is the *disjunction* of ϕ_1 and ϕ_2. Expression $(\phi_1 \wedge \phi_2)$ is the *conjunction* of ϕ_1 and ϕ_2. An expression involving one Boolean variable of the form x_i or $\neg x_i$ is a *literal*.

Note/Definition. Definition 4.1 gives the *syntax* of Boolean expression. The logical meaning is given by its *semantics* which yields the truth values of the expressions in terms of the truth values of the Boolean variables. We start with structurally simple expressions and then inductively build up to more structurally complicated expressions (by putting Boolean expressions into Boolean expressions) using Definition 4.1 parts (b), (c), and (d).

Definition 4.2. A *truth value* T is a mapping from a finite set X' of Boolean variables, $X' \subset X$, to the set of *truth values* $\{\mathbf{true}, \mathbf{false}\}$. Let ϕ be a Boolean expression. Define the set $X(\phi) \subset X$ of the *Boolean variables appearing in* ϕ inductively as follows: If ϕ is a Boolean variable x_i, then $X(\phi) = \{x_i\}$. If $\phi = \neg\phi_1$ then $X(\phi) = X(\phi_i)$. If $\phi = (\phi_1 \vee \phi_2)$, or if $\phi = (\phi_1 \wedge \phi_2)$, then $X(\phi) = X(\phi_1) \cup X(\phi_2)$. Next, let T be a truth assignment defined on a set X' of Boolean variables such that $X(\phi) \subset X'$; such a truth assignment is *appropriate to* ϕ. Suppose that T is appropriate to ϕ. We now define what it means for T to *satisfy* ϕ, denoted $T \models \phi$. If ϕ is a variable with $x_i \in X(\phi)$, then $T \models \phi$ if $T(x_i) = \mathbf{true}$. If $\phi = \neg\phi_1$, then $T \models \phi$ if $T \not\models \phi_1$ (that is, if it is not the case that $T \models \phi_1$). If $\phi = (\phi_1 \vee \phi_2)$, then $T \models \phi_1$ or $T \models \phi_2$. If $\phi = (\phi_1 \wedge \phi_2)$ then $T \models \phi$ if both $T \models \phi_1$ and $T \models \phi_2$ hold.

Example 4.1. Consider the Boolean expression $\phi = ((\neg x_1 \vee x_2) \wedge x_3)$. First, the set of Boolean variables X' must include the Boolean variables appearing in ϕ is (illustrating the inductive nature of Definition 4.2):

$$X(\phi) = X((\neg x_1 \vee x_2) \wedge x_3) = X(\neg x_1 \vee x_2) \cup X(x_3) = (X(\neg x_1) \cup X(x_2)) \cup X(x_3)$$

$$= X(x_1) \cup X(x_2) \cup X(x_3) = \{x_1\} \cup \{x_2\} \cup \{x_3\} = \{x_1, x_2, x_3\}.$$

So a truth assignment that is appropriate to ϕ is $T(x_1) = T(x_3) = \mathbf{true}$ and $T(x_2) = \mathbf{false}$. We address the the question: Does T satisfy ϕ (symbolically, does $T \models \phi$)? Since $x_3 \in X(\phi)$ and $T(x_3) = \mathbf{true}$ then by the definition of $\models$ we have $T \models x_3$. By the definition of "satisfaction" in the case of conjunction (that is, $\phi_1 \wedge \phi_2$ or, in this case, $((\neg x_1 \vee x_2) \wedge x_3))$ we need both $T \models (\neg x_1 \vee x_2)$ and $T \models x_3$. We also have $x_1, x_2 \in X(\phi) = \{x_1, x_2, x_3\}$, but $T(x_2) = \mathbf{false}$ and so

$T \not\models x_2$ and similarly, since $T(\neg x_1) = \textbf{false}$, then $T \not\models \neg x_1$. So $T \not\models (\neg x_1 \vee x_2)$ (since satisfaction in the case of disjunction requires that T satisfy one or the other of $\neg x_1$ and x_2). For T to satisfy the conjunction $\phi = (\neg x_1 \vee x_2) \wedge x_3$ requires T to satisfy both $\neg x_1 \vee x_2$ and x_3, so that we conclude that T does not satisfy ϕ; i.e., $T \not\models \phi$.

Definition. We denote $(\neq \phi_1 \vee \phi_2)$ as $(\phi_1 \Rightarrow \phi_2)$. We denote $(\phi_1 \Rightarrow \phi_2) \wedge (\phi_2 \Rightarrow \phi_1))$ as $(\phi_1 \Leftrightarrow \phi_2)$. Two expressions ϕ_1 and ϕ_2 are *equivalent*, denoted $\phi_1 \equiv \phi_2$, if for any truth assignment T appropriate to both of them, $T \models \phi_1$ if and only if $T \models \phi_2$.

Note. By considering truth tables, we see the motivation for the symbols $\Rightarrow$ ("implication") and $\Leftrightarrow$ ("biconditional"); see my online notes for Mathematical Reasoning (MATH 3000) on Section 1.3. Conditional Statements. In Problem 4.4.3 it is to be shown that $\phi_1 \equiv \phi_2$ is the same as saying that for any appropriate T, $T \models (\phi_1 \Leftrightarrow \phi_2)$. Next, we state some equivalences of Boolean expressions.

Proposition 4.1. Let ϕ_1, ϕ_2, and ϕ_3 be arbitrary Boolean expressions. Then:

(1) Commutativity of disjunction: $(\phi_1 \vee \phi_2) \equiv (\phi_2 \vee \phi_1)$.

(2) Commutativity of conjunction: $(\phi_1 \wedge \phi_2) \equiv (\phi_2 \wedge \phi_1)$.

(3) $\neg\neg\phi_1 \equiv \phi_1$.

(4) Associativity of disjunction: $((\phi_1 \vee \phi_2) \vee \phi_3) \equiv (\phi_1 \vee (\phi_2 \vee \phi_3))$.

(5) Associativity of conjunction: $((\phi_1 \wedge \phi_2) \wedge \phi_3) \equiv (\phi_1 \wedge (\phi_2 \wedge \phi_3))$.

(6) Distribution of conjunction over disjunction: $((\phi_1 \wedge \phi_2) \vee \phi_3) \equiv ((\phi_1 \vee \phi_3) \wedge (\phi_2 \vee \phi_3))$.

(7) Distribution of disjunction over conjunction: $((\phi_1 \vee \phi_2) \wedge \phi_3) \equiv ((\phi_1 \wedge \phi_3) \vee (\phi_2 \wedge \phi_3))$.

(8) De Morgan's Law for disjunction: $\neg(\phi_1 \vee \phi_2) \equiv (\neg\phi_1 \wedge \neg\phi_2)$.

(9) De Morgan's Law for conjunction: $\neg(\phi_1 \wedge \phi_2) \equiv (\neg\phi_1 \vee \neg\phi_2)$.

(10) Idempotency of disjunction: $\phi_1 \vee \phi_1 \equiv \phi_1$.

(11) Idempotency of conjunction: $\phi_1 \wedge \phi_1 \equiv \phi_1$.

Note. Notice that in any part of Proposition 4.1, we can interchange $\vee$ and $\wedge$ to get another part of Proposition 4.1. This is sometimes called "duality." Notice that each of these properties has an analogous property in set theory where we replace $\vee$ with $\cup$, $\wedge$ with $\cap$, and $\neg$ with set complement; see my online notes for Mathematical Reasoning (MATH 3000) on Section 2.5. Union, Intersection, and Complement (in particular, see Theorem 2.16). Proposition 4.1 also allows us to reorder a string of conjunctions and disjunctions, and to simplify expressions. For example, $(((x_1 \vee \neg x_3) \vee x_2) \vee x_2) \vee x_4 \vee (x_2 \vee x_5))$ is the same as $(x_1 \vee \neg x_3 \vee x_2 \vee x_4 \vee x_2 \vee x_5)$, which simplifies to $(x_1 \vee \neg x_3 \vee x_2 \vee x_4 \vee x_5)$. Notationally, we let $\bigwedge_{i=1}^n \phi_i$ denote $(\phi_1 \wedge \phi_2 \wedge \cdots \wedge \phi_n)$, and let $\bigvee_{i=1}^n \phi_i$ denote $(\phi_1 \vee \phi_2 \vee \cdots \vee \phi_n)$.

Definition. A Boolean expression ϕ is in *conjunctive normal form* if $\phi = \bigwedge_{i=1}^n C_i$, where $n \geq 1$, and each of the C_js is the disjunction of one or more literals (recall that a "literal" is an expression of the form x_i or $\neg x_i$). The C_js are the *clauses* of the expression. A Boolean expression ϕ is in *disjunctive normal form* if $\phi = \bigvee_{i=1}^{n} D_i$ where $n \geq 1$, and each of the D_js is the conjunction (that's right, CONjunction) of one of more literals. The D_js are the *implicants* of the expression in the disjunctive normal form.

Theorem 4.1. Every Boolean expression is equivalent to one in conjunctive normal form, and to one in disjunctive normal form.

Revised: 9/25/2022