CS304 – CS310: Mathematical and Logical Aspects of Computing

Lecture 2: Boolean Expressions

Friday, 7th Sep 2012

TODAY:

1. Boolean expressions; assignments and interpretations.
2. Logical equivalences; distributive laws; de Morgan's laws.
3. More 2-place operators: NAND, NOR.

Recall... Boolean Operators

A Boolean operator takes arguments of *Boolean type* (i.e., with one of the true values F or T assigned to them) and maps these to one of F or T.

If the operator takes n arguments, we say it is an *n-place Boolean operator*.

- There are **4** possible 1-place operators, of which **negation** is the most interesting. It is denoted by $\neg$, and read as "not".
- There are **16** possible 2-place operators. Among the most important are
  - *Conjunction*, written as $a \wedge b$, read as "and".
  - *Disjunction*, written as $a \vee b$, read as "or".
  - *Non-equivalence*, a.k.a., the *exclusive or* operator, and denoted as $\oplus$

Boolean expressions

The previous slide we used the symbols a and b as variables that could take the value F or T. Such variables are usually called "*Boolean variables*", "*propositional variables*", "*atomic propositions*" or, simply, "*atoms*".

We will denote this set of variables as $\mathcal{P}$. By writing $x \in \mathcal{P}$, we mean that x is a Boolean variable that can have the value F or T.

Alternative: use $\{0, 1\}$ for $\{F, T\}$, and write $x \in \mathbb{Z}_2$, where $\mathbb{Z}_2 = \{0, 1\}$. Other common notation is to use $\{\perp, \top\}$.

Definition (Boolean Expression / Propositional formula)

We can define the set of all **Boolean Expressions** (a.k.a. Formulas), denoted $\mathcal{F}$ the following, recursive manner:

1. F, T and $x_i \in \mathcal{P}$ are Boolean Expressions. That is, they belong to the set $\mathcal{F}$.
2. If E belongs to $\mathcal{F}$, then so too does $\neg E$.
3. If $\star$ is a 2-place Boolean operator, and E_1 and E_2 are both in *BEs*, then then expression $E_1 \star E_2$ is in *BEs*.

Boolean expressions — Assignments and interpretations

When we give a variable a particular value or F or T, we call this an "*assignment*". So every Boolean variable has two possible assignments.

When we give a particular assignment to arguments of a Boolean Expression, this is called an "*interpretation*". This means that every 2-place operator has **four** possible interpretations.

Examples:

....

Definition (Logically equivalent)

Two Boolean expressions $E_1, E_2 \in \mathcal{F}$ are *logically equivalent* if $v(E_1) = v(E_2)$ for all possible interpretations v. We write this as $E_1 \equiv E_2$. Informally, they are logically equivalent if their truth-table are the same.

Now that we have this definition we can investigate some important Boolean identities. **Examples:**

Distributive laws; de Morgan's Laws

Example

Determine if

$$(a \vee b) \wedge c \text{ is logically equivalent to } (a \wedge c) \vee (b \wedge c).$$

. . . .

de Morgan's laws

(a) $\neg(a \wedge b) \equiv \neg a \vee \neg b$

(b) $\neg(a \vee b) \equiv \neg a \wedge \neg b$

We will do (b) and leave (a) as an exercise.

More important 2-place operators

Earlier we looked at three 2-place operators ($\wedge$, $\vee$, $\oplus$). Of the remaining **thirteen**, several have special significance in Mathematics, Computing and Electronics. These are

1. *nor*, written as $a \downarrow b$;
2. *nand*, written as $a \uparrow b$;
3. *equivalence*, written as $a \leftrightarrow b$.
4. *implication*, written as $a \rightarrow b$, and read as "a implies b";

More important 2-place operators NOR and NAND

The logic table for "NOR" is:

It is important to note how it can be written in term of OR and NOT. So we say that

$$a \downarrow b \text{ is } \textit{logically equivalent} \text{ to } \neg(a \vee b).$$

More important 2-place operators NOR and NAND

The logic table for "NAND" is:

Again, this can be written in term of AND and NOT. So:

$$a \uparrow b \text{ is } \textit{logically equivalent} \text{ to } \neg(a \wedge b).$$

. . . .

Exercises

1. Determine if $(a \wedge b) \vee c$ is logically equivalent to $(a \vee c) \wedge (b \vee c)$.
2. Show that $\neg(a \wedge b) \equiv \neg a \vee \neg b$.