

Propositional Calculus

- *Semantics* (1/3)

Moonzoo Kim
CS Division of EECS Dept.
KAIST

moonzoo@cs.kaist.ac.kr
<http://pswlab.kaist.ac.kr/courses/cs402-07>

Overview

- 2.1 Boolean operators
- 2.2 Propositional formulas
- 2.3 Interpretations

Boolean Operators

- A proposition (p, q, r, \dots) in a propositional calculus can get a boolean value (i.e. true or false)
- Propositional formula can be built by combining smaller formula with boolean operators such as \neg, \wedge, \vee
- How many different unary boolean operators exist?

p	o_1	o_2	o_3	o_4	\dots
T	T	T	F	F	
F	T	F	T	F	

- How many different binary boolean operators exist?

Binary Boolean Operators

x_1	x_2	\circ_1	\circ_2	\circ_3	\circ_4	\circ_5	\circ_6	\circ_7	\circ_8
<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>
<i>T</i>	<i>F</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>F</i>	<i>F</i>	<i>F</i>	<i>F</i>
<i>F</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>T</i>	<i>F</i>	<i>F</i>
<i>F</i>	<i>F</i>	<i>T</i>	<i>F</i>	<i>T</i>	<i>F</i>	<i>T</i>	<i>F</i>	<i>T</i>	<i>F</i>

x_1	x_2	\circ_9	\circ_{10}	\circ_{11}	\circ_{12}	\circ_{13}	\circ_{14}	\circ_{15}	\circ_{16}
<i>T</i>	<i>T</i>	<i>F</i>	<i>F</i>	<i>F</i>	<i>F</i>	<i>F</i>	<i>F</i>	<i>F</i>	<i>F</i>
<i>T</i>	<i>F</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>F</i>	<i>F</i>	<i>F</i>	<i>F</i>
<i>F</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>T</i>	<i>F</i>	<i>F</i>
<i>F</i>	<i>F</i>	<i>T</i>	<i>F</i>	<i>T</i>	<i>F</i>	<i>T</i>	<i>F</i>	<i>T</i>	<i>F</i>

Boolean Operators

op	name	symbol	op	name	symbol
o_2	disjunction	\vee	o_{15}	nor	\downarrow
o_8	conjunction	\wedge	o_9	nand	\uparrow
o_5	implication	\rightarrow	o_{12}		
o_3	reverse implication	\leftarrow	o_{14}		
o_7	equivalence	\leftrightarrow	o_{10}	exclusive or	\oplus

Boolean Operators

- The first five binary operators can all be defined in terms of any one of them plus **negation**
- Nand or nor by itself is sufficient to define all other operators.
- The choice of an interesting set of operators depends on the application
 - Mathematics is generally interested in one-way logical deduction (given a set of axioms, what do they imply?).
 - So implication together with negation are chosen as the basic operators

Propositional formulas

- Def 2.1 A formula $fml \in \mathcal{F}$ in the propositional calculus is a word that can be derived from the following grammar, starting from the initial non-terminal fml
 1. $fml ::= p$ for any $p \in \mathcal{P}$
 2. $fml ::= \neg fml$
 3. $fml ::= fml \text{ op } fml$ where $op \in \{ \vee, \wedge, \rightarrow, \leftarrow, \leftrightarrow, \downarrow, \uparrow, \oplus \}$
- Each derivation of a formula from a grammar can be represented by a **derivation tree** that displays the application of the grammar rules to the non-terminals
 - non-terminals: symbols that occur on the left-hand side of a rule
 - terminal: symbols that occur on only the right-hand side of a rule
- From the derivation tree we can obtain a **formation tree**
 - by replacing an fml non-terminal by the child that is an **operator** or an **atom**

Ambiguous representation of formulas

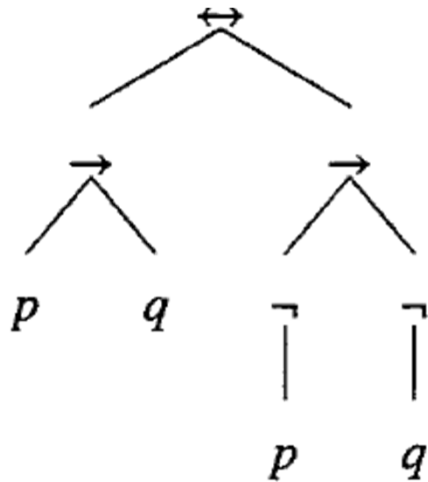


Figure 2.3 Formation tree for $p \rightarrow q \leftrightarrow \neg p \rightarrow \neg q$

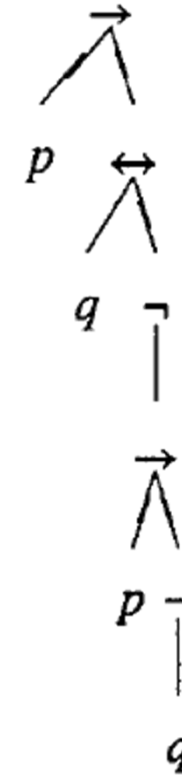


Figure 2.4 Another formation tree

Formulas created by a Polish notation

- There will be no ambiguity if the linear sequence of symbols is created by a preorder traversal of the formal tree
 - Visit the root, visit the left subtree, visit the right subtree
- $\leftrightarrow \rightarrow p q \rightarrow \neg p \neg q$
- $\rightarrow p \leftrightarrow q \neg \rightarrow \neg p \neg q$
- Polish notation is used in the internal representation of an expression in a computer
 - advantage: the expression can be executed in the linear order the symbols appear
- If we rewrite the first formula from backwards
 - $q \neg p \neg \rightarrow qp \rightarrow \leftrightarrow$
 - can be directly compiled to the following sequence of instructions
 - Load q
 - Negate
 - Load p
 - Negate
 - Imply
 - load q
 - Load p
 - Imply
 - Equiv

Other ways to remove ambiguity

- Use parenthesis
- Define precedence and associativity
 - The precedence order
 - $\neg > \wedge > \uparrow > \vee > \downarrow > \rightarrow > \leftrightarrow$
 - Operators are assumed to associate to the right
 - $a \rightarrow b \rightarrow c$ means $(a \rightarrow (b \rightarrow c))$
 - $a \vee b \vee c$ means $(a \vee (b \vee c))$
 - Some textbook considers $\wedge, \vee, \leftrightarrow$ as associate to the left. So be careful.

Structural induction

- Theorem 2.5. To show $\text{property}(A)$ for **all** formulas $A \in \mathcal{F}$, it suffices to show:
 - base case:
 - $\text{property}(p)$ for all atoms $p \in \mathcal{P}$
 - induction step:
 - Assuming $\text{property}(A)$, the $\text{property}(\neg A)$ holds
 - Assuming $\text{property}(A_1)$ and $\text{property}(A_2)$, then $\text{property}(A_1 \text{ op } A_2)$ hold, for each of the binary operators
- Example
 - Prove that **every** propositional formula can be equivalently expressed using only \uparrow

Interpretations

- Def 2.6 An **assignment** ν is a function $\nu: \mathcal{P} \rightarrow \{T, F\}$
 - that is ν assigns one of the truth values T or F to every atom
 - From now on we use two new syntax terms, “true” and “false”
 - $\text{fml} ::= \text{true} \mid \text{false}$ where $\nu(\text{true}) = T$ and $\nu(\text{false}) = F$
 - note that we need to distinguish “true” from T and “false” from F
 - “true” and “false” are syntactic terms in propositional logic, but T and F are truth values
- Note that an assignment ν can be **extended** to a function $\nu: \mathcal{F} \rightarrow \{T, F\}$, mapping formulas to truth values by the inductive definition.
 - ν is called an **interpretation**

Interpretations

- Inductive truth value calculation for given formula A

A	$v(A_1)$	$v(A_2)$	$v(A)$
$\neg A_1$	T		F
$\neg A_1$	F		T
$A_1 \vee A_2$	F	F	F
$A_1 \vee A_2$	otherwise		T
$A_1 \wedge A_2$	T	T	T
$A_1 \wedge A_2$	otherwise		F
$A_1 \rightarrow A_2$	T	F	F
$A_1 \rightarrow A_2$	otherwise		T

A	$v(A_1)$	$v(A_2)$	$v(A)$
$A_1 \uparrow A_2$	T	T	F
$A_1 \uparrow A_2$	otherwise		T
$A_1 \downarrow A_2$	F	F	T
$A_1 \downarrow A_2$	otherwise		F
$A_1 \leftrightarrow A_2$	$v(A_1) = v(A_2)$		T
$A_1 \leftrightarrow A_2$	$v(A_1) \neq v(A_2)$		F
$A_1 \oplus A_2$	$v(A_1) \neq v(A_2)$		T
$A_1 \oplus A_2$	$v(A_1) = v(A_2)$		F

Figure 2.5 Evaluation of truth values of formulas

- Theorem 2.9 An assignment can be extended to **exactly one** interpretation
- Theorem 2.10 Let $\mathcal{P}' = \{p_{i_1}, \dots, p_{i_n}\} \subseteq \mathcal{P}$ be the atoms appearing in $A \in \mathcal{F}$. Let ν_1 and ν_2 be assignments that agree on \mathcal{P}' , that is $\nu_1(p_{i_k}) = \nu_2(p_{i_k})$ for all $p_{i_k} \in \mathcal{P}'$. Then the interpretations agree on A , that is $\nu_1(A) = \nu_2(A)$.

Examples

Example 2.7 Let $A = (p \rightarrow q) \leftrightarrow (\neg q \rightarrow \neg p)$, and let ν the assignment such that $\nu(p) = F$ and $\nu(q) = T$, and $\nu(p_i) = T$ for all other $p_i \in \mathcal{P}$. Extend ν to an interpretation. The truth value of A can be calculated inductively using Figure 2.5:

$$\nu(p \rightarrow q) = T$$

$$\nu(\neg q) = F$$

$$\nu(\neg p) = T$$

$$\nu(\neg q \rightarrow \neg p) = T$$

$$\nu((p \rightarrow q) \leftrightarrow (\neg q \rightarrow \neg p)) = T.$$

Example 2.8 $\nu(p \rightarrow (q \rightarrow p)) = T$ but $\nu((p \rightarrow q) \rightarrow p) = F$ under the above interpretation, emphasizing that the linear string $p \rightarrow q \rightarrow p$ is ambiguous. \square

Example 2.12 Let $S = \{p \rightarrow q, p, p \vee s \leftrightarrow s \wedge q\}$, and let ν be the assignment given by $\nu(p) = T$, $\nu(q) = F$, $\nu(r) = T$, $\nu(s) = T$. ν is an interpretation for S and assigns the truth values