University of Chinese Academy of Sciences

CS101

Logic Thinking

Propositional Logic: Properties and Examples

zxu@ict.ac.cn
zhangjialin@ict.ac.cn

Outline

- Foundation of logic
  - Propositional Logic
  - Predicative Logic
- Automata and Turing Machines
- Power and Limitation of Computing
  - Mechanical Theorem Proving
  - Church-Turing Hypothesis

Boolean algebra

- When viewing Boolean expressions and operators as an algebra, we have Boolean algebra
  - Similar but different from school algebra
  - Take the example when x=2, y=3, z=5 for school algebra

Boolean Algebra	School Algebra
(x•y)•z = x•(y•z)	(2•3)•5 = 2•(3•5)
(x+y)+z = x+(y+z)	(2+3)+5 = 2+(3+5)
x•y = y•x	2•3 = 3•2
x+y = y+x	2+3 = 3+2
(x+y)•z = (x•z)+(y•z)	(2+3)•5 =2•5+3•5
(x•y)+z = (x+z)•(y+z)	(2•3)+5 ≠(2+5)•(3+5)
x+0 = x, x•1 = x	2+0=2, 2•1=2
x•0 = 0, x+1 = 1	2•0=0, 2+1≠1
x•x = x, x+x = x	2•2 ≠ 2, 2+2 ≠ 2
(x•y)+x = x, (x+y)•x = x	(2•3)+2≠2, (2+3)•2≠2

Properties (Boolean algebra axioms)

- Identity
  - $x \vee 0 = x, \; x \vee 1 = 1$
- Annihilator
  - $x \wedge 0 = 0, \;\; x \wedge 1 = x$
- Complementation
  - $x \vee \neg x = 1, \; x \wedge \neg x = 0$
- Commutativity
  - $x \wedge y = y \wedge x, \;\; x \vee y = y \vee x, \;\; x \bigoplus y = y \bigoplus x$
- Associativity
  - $(x \wedge y) \wedge z = x \wedge (y \wedge z), \;\; (x \vee y) \vee z = x \vee (y \vee z)$

Properties (2)

- Distributivity
  - $(x \wedge y) \vee z = (x \vee z) \wedge (y \vee z), (x \vee y) \wedge z = (x \wedge z) \vee (y \wedge z)$
- De Morgan law
  - $\neg(x \vee y) = \neg x \wedge \neg y$
  - $\neg(x \wedge y) = \neg x \vee \neg y$
- Implication
  - $x \rightarrow y = \neg x \vee y$
  - $x \rightarrow y = \neg y \rightarrow \neg x$ (original = contrapositive)
- Xor
  - $x \oplus y = (\neg x \wedge y) \vee (x \wedge \neg y)$
  - $x \oplus y = (x \vee y) \wedge (\neg x \vee \neg y)$

Normal Form

- Any Boolean function can be uniquely expressed
  - In Conjunctive Normal Form
    - a special form of Boolean expression, or
  - In Disjunctive Normal Form
    - another special form of Boolean expression

- Conjunctive Normal Form, CNF
  - $f(x_1, \ldots, x_n) = \ Q_1 \land Q_2 \land Q_3 \land \cdots \land Q_m$
  - where $Q_i = l_1 \lor l_2 \lor \cdots \lor l_n$, $\quad l_j = x_j$ or $\neg x_j$

- Disjunctive Normal Form, DNF
  - $f(x_1, \ldots, x_n) = \ Q_1 \lor Q_2 \lor Q_3 \lor \cdots \lor Q_m$
  - where $Q_i = l_1 \land l_2 \land \cdots \land l_n$, $\quad l_j = x_j$ or $\neg x_j$

- Example: $x \rightarrow (y \rightarrow z) \ = ?$

x	y	z	f
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

  - CNF: $(\neg x \vee \neg y \vee z)$
  - DNF: $(\neg x \wedge \neg y \wedge \neg z) \vee (\neg x \wedge \neg y \wedge z) \vee (\neg x \wedge y \wedge \neg z) \vee (\neg x \wedge y \wedge z) \vee (x \wedge \neg y \wedge \neg z) \vee (x \wedge \neg y \wedge z) \vee (x \wedge y \wedge z)$

Examples of Boolean expressions

- Boolean algebra axioms can be used to reduce equivalent expressions into one expression
  - E.g., x, $x \cdot x$, $x \cdot x \cdot x$, ... , $x + x$, $x + x + x$, ... are all equivalent to x
- Then, for any *n*, the set of Boolean expressions *L* is finite and unique
- When *n* = 1, $L = \{0,\ 1,\ x,\ \bar{x}\}$

When n=2, L can be obtained in 3 rounds

- Round 1: L contains constants, variables, and their negations. There are six expressions $0, 1, x_1, x_2, \overline{x_1}, \overline{x_2}$

x_1	x_2	y
0	0	**0**
0	1	**0**
1	0	**0**
1	1	**0**

x_1	x_2	y
0	0	**1**
0	1	**1**
1	0	**1**
1	1	**1**

x_1	x_2	y
0	0	**0**
0	1	**0**
1	0	**1**
1	1	**1**

x_1	x_2	y
0	0	**0**
0	1	**1**
1	0	**0**
1	1	**1**

x_1	x_2	y
0	0	**1**
0	1	**1**
1	0	**0**
1	1	**0**

x_1	x_2	y
0	0	**1**
0	1	**0**
1	0	**1**
1	1	**0**

- Round 2: L has eight new expressions $\overline{x_1} + \overline{x_2},\ \overline{x_1} + x_2,\ x_1 + \overline{x_2},\ x_1 + x_2\ \ \overline{x_1} \cdot \overline{x_2},\ \overline{x_1} \cdot x_2,\ \ x_1 \cdot \overline{x_2},\ x_1 \cdot x_2$

x_1	x_2	y
0	0	**1**
0	1	**1**
1	0	**1**
1	1	**0**

x_1	x_2	y
0	0	**1**
0	1	**1**
1	0	**0**
1	1	**1**

x_1	x_2	y
0	0	**1**
0	1	**0**
1	0	**1**
1	1	**1**

x_1	x_2	y
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**1**

x_1	x_2	y
0	0	**1**
0	1	**0**
1	0	**0**
1	1	**0**

x_1	x_2	y
0	0	**0**
0	1	**1**
1	0	**0**
1	1	**0**

x_1	x_2	y
0	0	**0**
0	1	**0**
1	0	**1**
1	1	**0**

x_1	x_2	y
0	0	**0**
0	1	**0**
1	0	**0**
1	1	**1**

- Round 2: L has two new expressions $\overline{x_1} \cdot \overline{x_2} + x_1 \cdot x_2,\ \overline{x_1} \cdot x_2 + x_1 \cdot \overline{x_2}$

x_1	x_2	y
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**0**

x_1	x_2	y
0	0	**1**
0	1	**0**
1	0	**0**
1	1	**1**

Boolean Function

- How many different n-input-1-output Boolean functions are there?
  - Equivalent functions are counted as one function
    - Functions having the same truth table are equivalent functions

  - $n = 0$: 2 (0 and 1)
  - $n = 1$: 4 ($0, 1, x$ and $\neg x$)
  - $n = 2$: ?

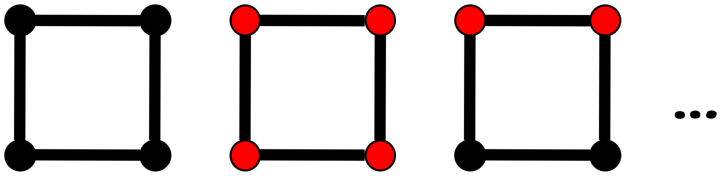
- 2^{2^n}

- Why?

x_1	x_2	…	x_{n-1}	x_n	y
0	0	…	0	0	0 or 1
0	0	…	0	1	0 or 1
0	0	…	1	0	0 or 1
0	0	…	1	1	0 or 1
…	…	…	…	…	0 or 1
1	1	…	1	0	0 or 1
1	1	…	1	1	0 or 1

Thinking Problem: Boolean Function

- How many different n-input 1-output monotone Boolean functions are there?
- Monotone Boolean function
  - for any $(x_1, \dots, x_n), (y_1, \dots, y_n)$, if $\forall i, x_i \leq y_i$, we have $f(x_1, \dots, x_n) \leq f(y_1, \dots, y_n)$
  - $n = 0$: 2 (0 and 1)
  - $n = 1$: 3 $(0, 1, x)$
  - $n = 2$: 6
  - $n = 3$: ?



The congruent triangle problem

- Consider the following propositions
  - *P*: "two triangles are congruent"; Q: "two triangles are similar"
  - Original: $P \to Q$.
  - If two are congruent triangles, they are similar.
  - Inverse: $\neg P \to \neg Q$

    If two are not congruent triangles, they are not similar.
  - Converse: $Q \to P$

    If two are similar triangles, they are congruent.
  - Contrapositive: $\neg Q \to \neg P$

    If two are not similar triangles, they are not congruent.
- A proposition and its contrapositive are equivalent.
- The inverse and the converse are equivalent.
- Why?

The congruent triangle problem

- Let us prove
  - "The inverse and the converse are equivalent"
  - Formally, $\neg P \to \neg Q = Q \to P$
- Proof process

• $(\neg P) \to (\neg Q)$	The given inverse statement
• $= \neg(\neg P) \vee (\neg Q)$	by implication property $P \to Q = \neg P \vee Q$
• $= P \vee (\neg Q)$	eliminate double negation
• $= \neg Q \vee P$	use communicative law
• $= Q \to P$	obtain the converse by implication property

- The proof does not involve any Trigonometry knowledge
- Prove "a proposition and its contrapositive are equivalent"

The impatient guide problem

- Problem context
  - A tourist is traveling in the land of Oz and wants to go to the Emerald City. The tourist reaches a crossroad with paths P and Q, one of which leads to the Emerald City. There is a guide G at the crossroad, who comes from either the Honest Village or the Lying Village. Anyone from the Honest Village always tells the truth, and anyone from the Lying Village always tells lies. The guide is impatient, in that G only answers one question from the tourist, and the answer is either "Yes" or "No".
- Q: What question should the tourist ask the guide, to determine the correct path?

- The question asked:
    - Are your answers the same, to the two questions "are you from the Honest Village" and "does path P lead to the Emerald City"?
- If the answer is "Yes", take path P; if the answer is "No", take path Q.
- The question becomes $\neg(H \oplus S)$, assuming the following notations
    - H denotes the proposition "G is from the Honest Village". That is, H=1 means G is from the Honest Village; H=0 means G is from the Lying Village.
    - S denotes the proposition "Path P leads to the Emerald City". That is, S=1 means path P leads to the Emerald City; S=0 means path Q leads to the Emerald City.

H	S	$\neg(H \oplus S)$	Comments
0	0	1	G is lying and the true value of the question is "Yes". The answer is "No", take path Q.
0	1	0	G is lying and the true value of the question is "No". The answer is "Yes", take path P.
1	0	0	G is telling the truth and the true value of the question is "No". The answer is "No", take path Q.
1	1	1	G is telling the truth and the true value of the question is "Yes". The answer is "Yes", take path P.

The adder implementation problem

- Realizing addition of two *n*-bit numbers with Boolean logic operators
- Design an adder, which takes two *n*-bit numbers *X* and *Y* as inputs and produce an *n*-bit number *Z* as the output
  - $(x_n \ldots x_1)_2 + (y_n \ldots y_1)_2 + c_0 = (c_n z_n \ldots z_1)_2$
- Here,
  - c_0 is the least significant carry bit
  - c_n is the most significant carry bit
- Only Boolean logic operators can be used
  - What are Boolean logic operators?

Boolean expressions of the output: Z

- When n=1, we have
  - a 1-bit full-adder $(c_{out}z)_2 = x + y + c_{in}$
  - 1 result bit $z = x \oplus y \oplus c_{in}$
  - 1 carry-out bit $c_{out} = (x \wedge y) \vee ((x \oplus y) \wedge c_{in})$
- For n>1, we cascade multiple full adders to obtain the n-bit adder
  - $z_1 = x_1 \oplus y_1 \oplus c_0;\quad c_1 = (x_1 \wedge y_1) \vee ((x_1 \oplus y_1) \wedge c_0)$
  - $z_2 = x_2 \oplus y_2 \oplus c_1;\quad c_2 = (x_2 \wedge y_2) \vee ((x_2 \oplus y_2) \wedge c_1)$
  - $z_3 = x_3 \oplus y_3 \oplus c_2;\quad c_3 = (x_3 \wedge y_3) \vee ((x_3 \oplus y_3) \wedge c_2)$
  - …………
  - $z_n = x_n \oplus y_n \oplus c_{n-1};\quad c_n = (x_n \wedge y_n) \vee ((x_n \oplus y_n) \wedge c_{n-1})$

Two examples helpful to Text Hider

- Go through the two examples independently
- Example 19. The parity program to show logic and bit-shift operations
- Example 20. Program to hide a character in a byte array