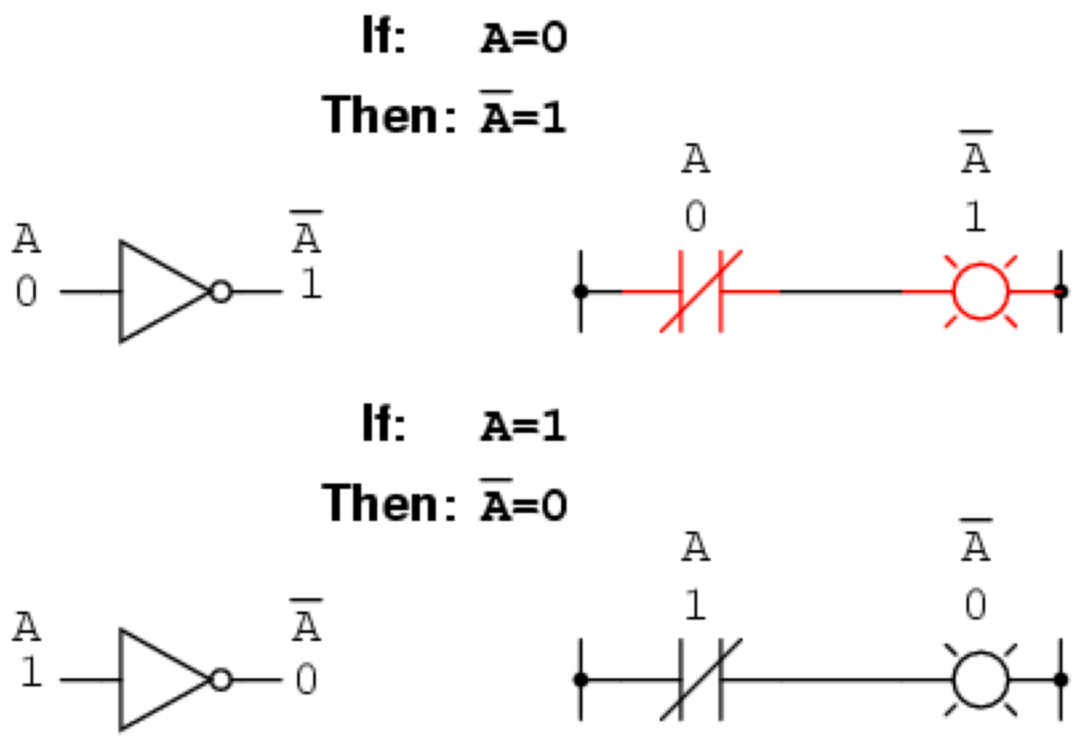
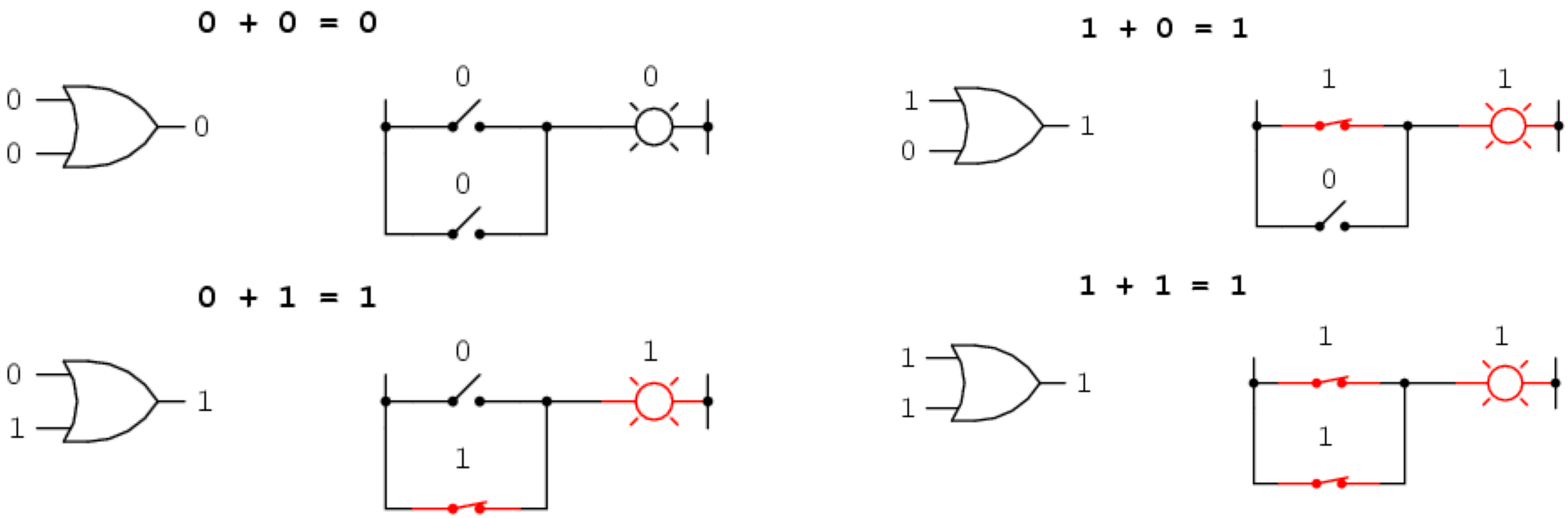
Applications of Logic

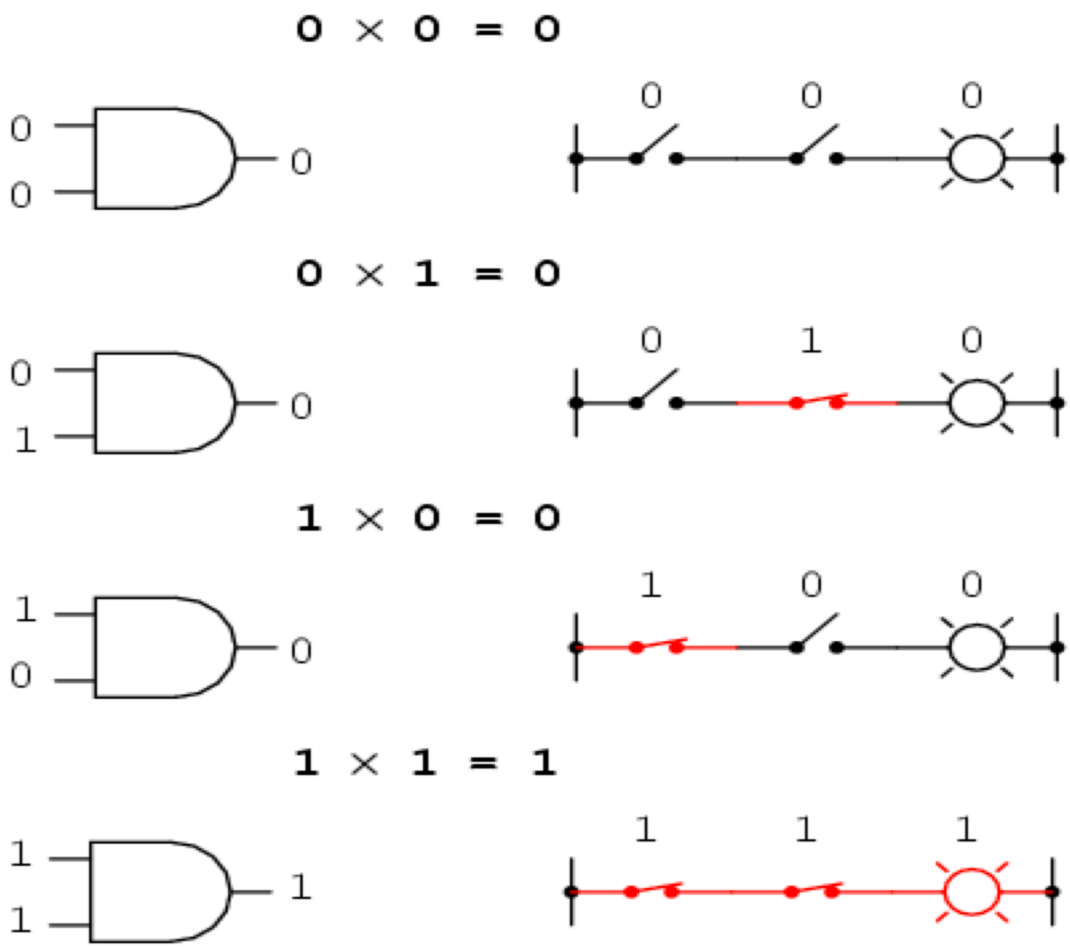
- Logical circuits are built using ¬, ∧, ∨ gates.
- NOT (¬) gate



OR (v) gate

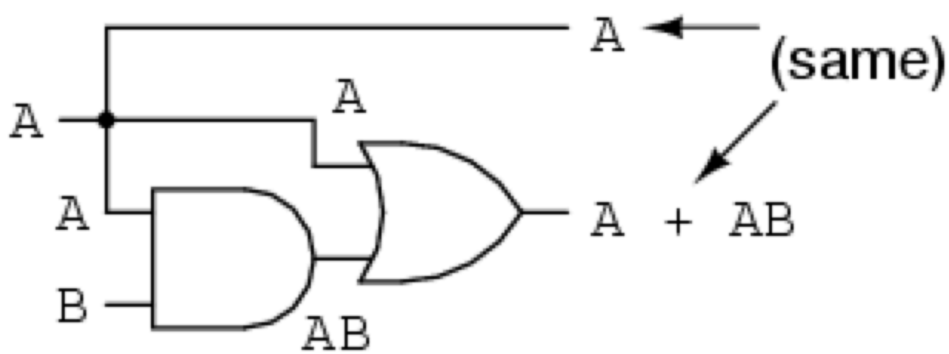


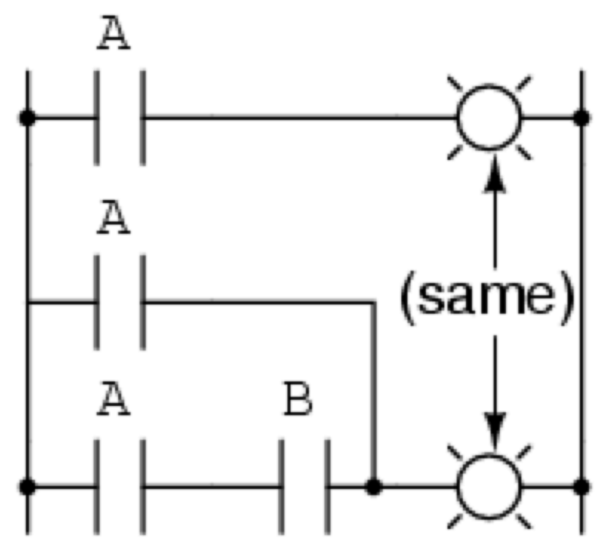
AND (∧) gate



Boolean Simplification

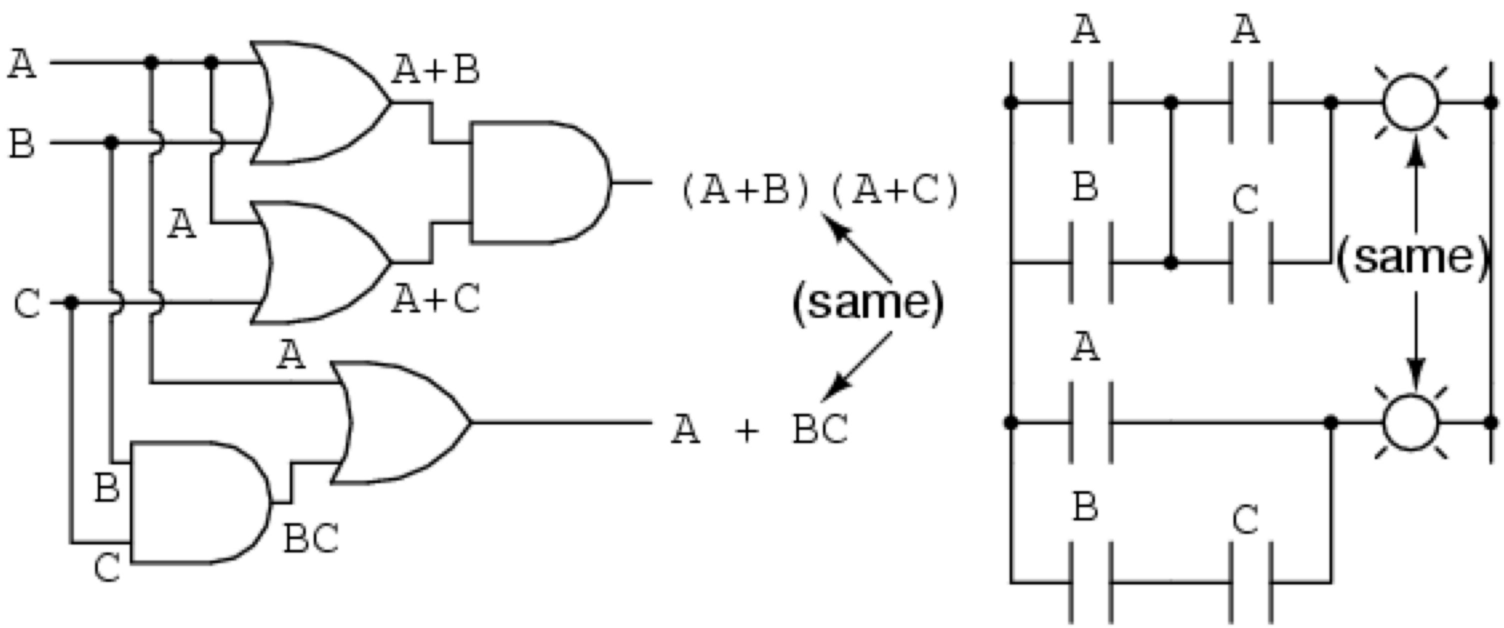
$$A + AB = A$$





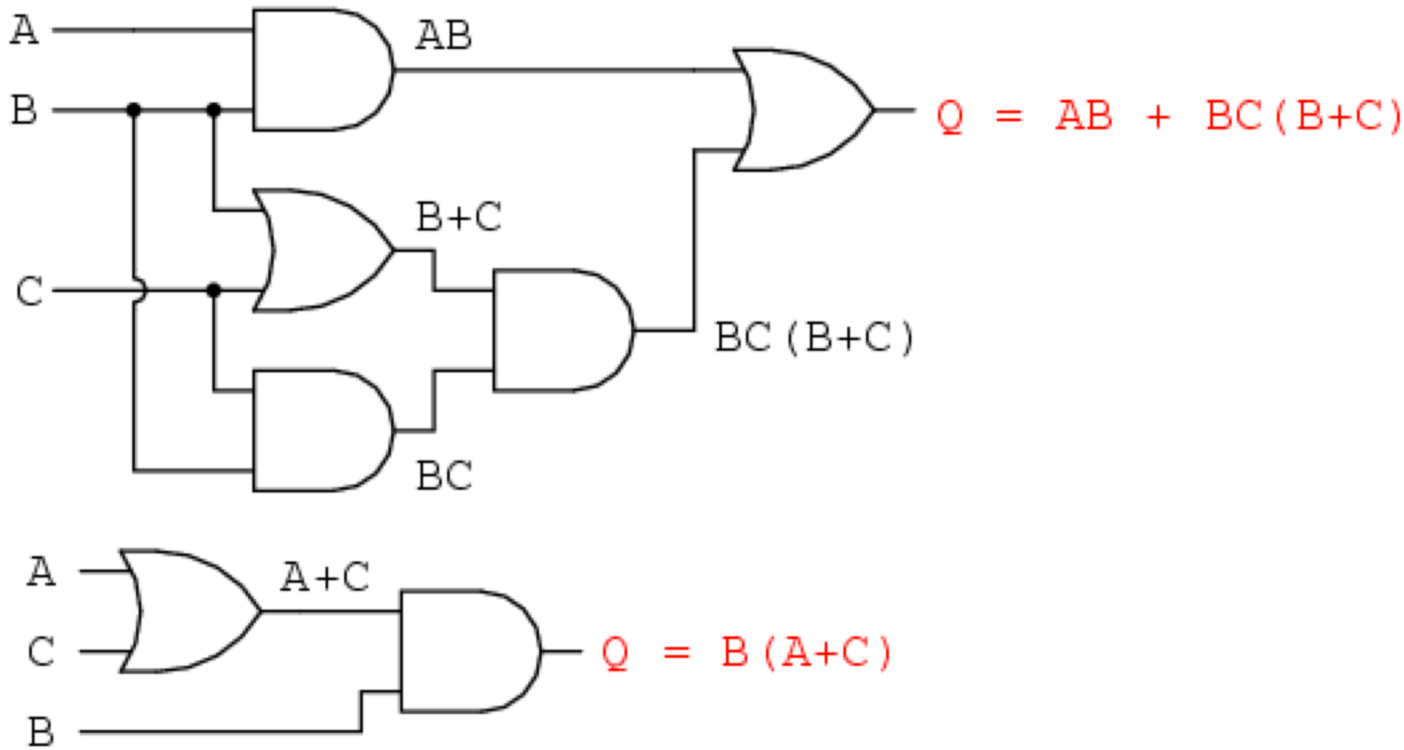
Boolean Simplification

(A + B)(A + C) = A + BC

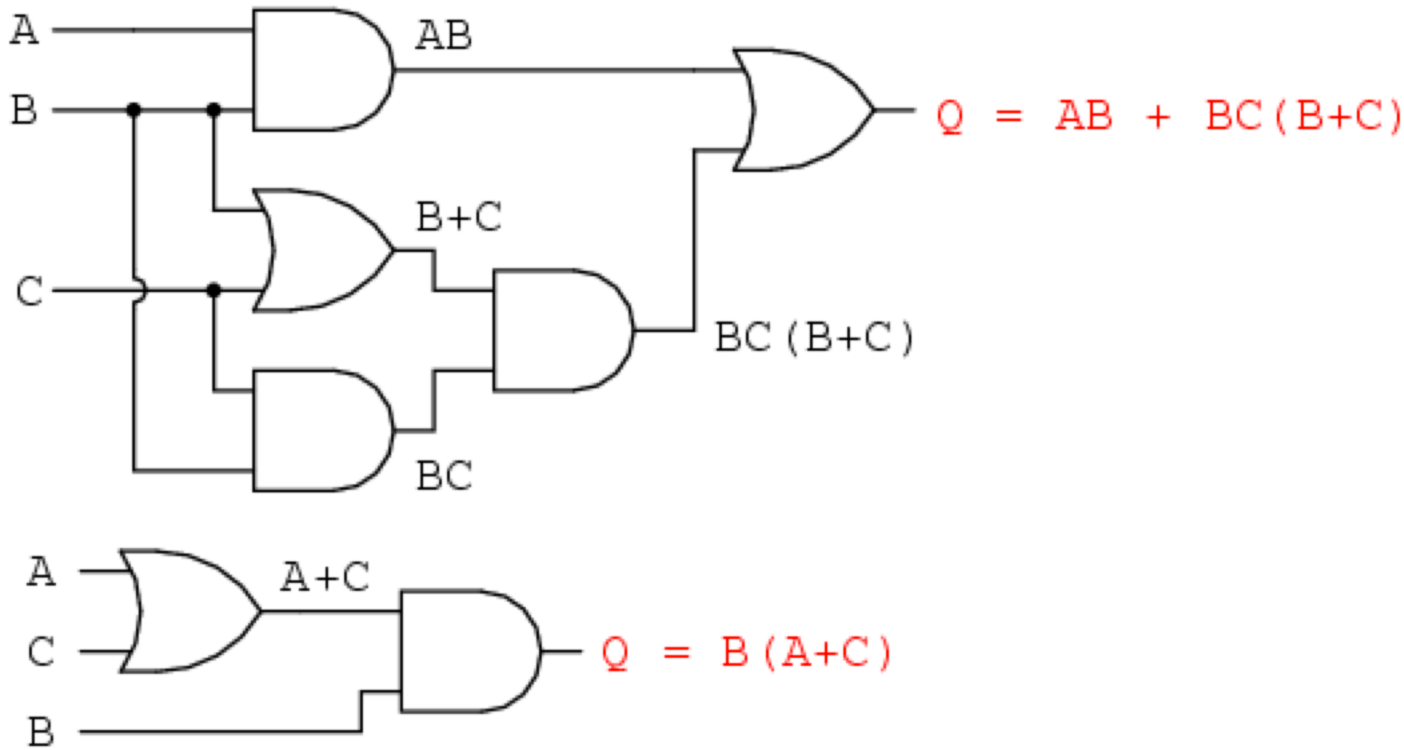


Distributive Law: $p \vee (q \wedge r) = (p \vee q) \wedge (p \vee r)$

Boolean Simplification (two equivalent circuits)



Boolean Simplification
(two equivalent circuits)



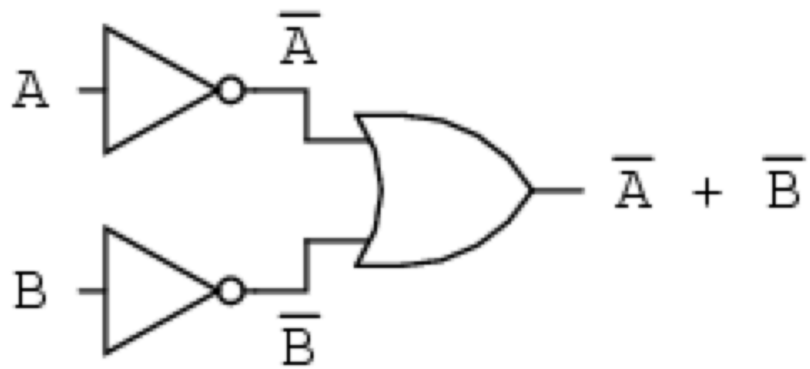
$(A \wedge B) \vee (B \wedge C) \wedge (B \vee C) = (A \wedge B) \vee ((B \wedge C) \wedge B) \vee ((B \wedge C) \wedge B)$

$= (A \wedge B) \vee (B \wedge C) \vee (B \wedge C)$

$= (A \wedge B) \vee (B \wedge C) = B \wedge (A \vee C)$

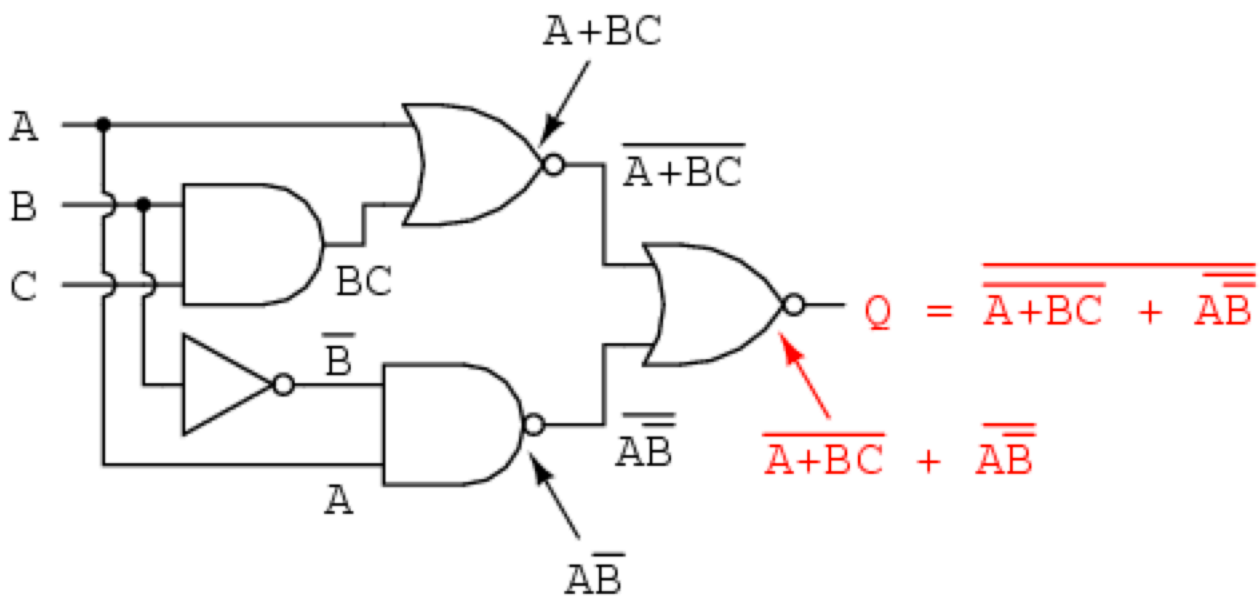
DeMorgan’s Law

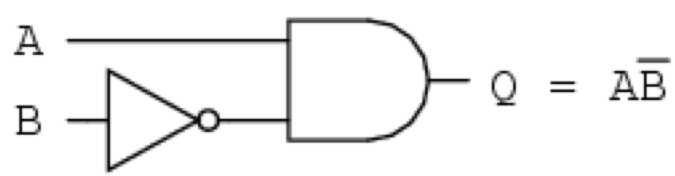
. . . is equivalent to . . .



$$\overline{AB} = \overline{A} + \overline{B}$$

Boolean Simplification
(two equivalent circuits)





Automated Theorem Proving

- Deals with the development of computer programs that show that some statement is a logical consequence of a set of statements.

An example

- Determine whether the following argument is valid.
  - She is a math major or cs major
  - If she doesn't know discrete math, she is not a math major.
  - If she knows discrete math, she is smart.
  - She is not a cs major.
- $\Rightarrow$ She is smart.

An example

- p: She is a math major
- q: She is a cs major
- r : She knows discrete math
- s : She is smart

1. $p \vee q$ is true
2. $\neg r \Longrightarrow \neg p$ is true
3. $r \Longrightarrow s$ is true
4. $\neg q$ is true

---

5. Therefore, s

Logical Inference (section 2.11)

- Suppose we know that the statement $p \Rightarrow q$ is true.
  - This tells us whenever p is true, q is true.
  - It does not tell us whether p or q is true. (They both could be false, or p is false and q is true.)
- Suppose in addition we know that p is true.
- In this case we can infer that q is true.
- This is called **logical inference.**

Rules of Inference

Rules of inference	Name
$p \Rightarrow q$ p $\therefore q$	Modus ponens
$p \Rightarrow q$ $\neg q$ $\therefore \neg p$	Modus tollens
$p \Rightarrow q$ $q \Rightarrow r$ $\therefore p \Rightarrow r$	Transitivity (hypothetical syllogism)
$p \vee q$ $\neg q$ $\therefore p$	Elimination (disjunctive syllogism)
$p \wedge q$ $\therefore p$	Simplification
p q $\therefore p \wedge q$	Conjunction
$\neg p \Rightarrow F$ $\therefore p$	Contradiction

Theoretical Computer Science

- Problem SAT (Satisfiability problem)
  - Given a formula involving n boolean variables, determine if it is possible to make the formula true by assigning truth values to the propositions.

Theoretical Computer Science

- A Boolean variable is a variable that can have a value 1 or 0. Thus, Boolean variable is a proposition.
- A term is a Boolean variable
- A literal is a term or its negation
- A clause is a disjunction of literals
- A sentence in PL is a conjunction of clauses
- Example of a formula
  - $(a \vee b \vee \neg c \vee \neg d) \wedge (\neg b \vee c) \wedge (\neg a \vee c \vee d)$
- A formula is satisfiable iff
  - we can assign a truth value
  - to each Boolean variables
  - such that the sentence evaluates to true (i.e., holds)

Theoretical Computer Science

- A Boolean variable is a variable that can have a value 1 or 0. Thus, Boolean variable is a proposition.
- A term is a Boolean variable
- A literal is a term or its negation
- A clause is a disjunction of literals
- A sentence in PL is a conjunction of clauses
- Example of a formula
  - $(a \vee b \vee \neg c \vee \neg d) \wedge (\neg b \vee c) \wedge (\neg a \vee c \vee d)$
- A formula is satisfiable iff
  - we can assign a truth value
  - to each Boolean variables
  - such that the sentence evaluates to true (i.e., holds)

This problem is extremely hard to solve