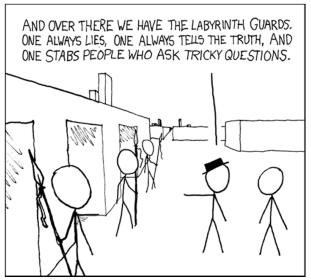
CSE 311



Foundations of Computing I

Fall 2014

Administrivia

Course Web: http://www.cs.washington.edu/311

Office Hours: 10 hours available; check web

Homework #1: Posted. Turn in (stapled) at the **start** of class on Wednesday (Oct 1)

Extra Credit: Not required to get a 4.0.
Counts separately.
In total, may raise grade by ~0.1

Last Time: Logical Connectives

p	$\neg p$
T	F
F	T

NOT

p	q	$p \wedge q$
T	T	T
T	F	F
F	T	F
F	F	F

AND

p	q	$p \vee q$
T	T	T
T	F	T
F	T	T
F	F	F

OR

p	q	$p \oplus q$
T	T	F
T	F	T
F	T	T
F	F	F

XOR

Last Time: $p \rightarrow q$

- "If p, then q" is a **promise**:
  - Whenever p is true, then q is true
  - Ask "has the promise been broken"

p	q	$p \rightarrow q$
F	F	T
F	T	T
T	F	F
T	T	T

Last Time: $p \rightarrow q$

- "If p, then q" is a **promise**:
  - Whenever p is true, then q is true
  - Ask "has the promise been broken"

p	q	$p \rightarrow q$
F	F	T
F	T	T
T	F	F
T	T	T

If it's raining, then I have my umbrella.

Last Time: Related Implications

- Implication: $p \rightarrow q$
- Converse: $q \rightarrow p$
- Contrapositive: $\neg q \rightarrow \neg p$
- Inverse: $\neg p \rightarrow \neg q$

How do these relate to each other?

Last Time: $p \leftrightarrow q$

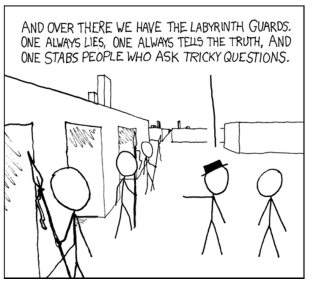
- *p* iff *q*
- *p* is equivalent to *q*
- *p* implies *q* and *q* implies *p*

p	q	$p \leftrightarrow q$
T	T	T
T	F	F
F	T	F
F	F	T

CSE 311: Foundations of Computing

Fall 2014

Lecture 2: Digital Circuits & More Logic



Digital Circuits

Computing With Logic

- **T** corresponds to 1 or "high" voltage
- **F** corresponds to 0 or "low" voltage

Gates

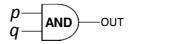
- Take inputs and produce outputs (functions)
- Several kinds of gates
- Correspond to propositional connectives (most of them)

And Gate

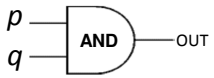
AND Connective vs. **AND Gate**

$p \wedge q$

p	q	$p \wedge q$
T	T	T
T	F	F
F	T	F
F	F	F



p	q	OUT
1	1	1
1	0	0
0	1	0
0	0	0



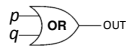
"block looks like D of AND"

Or Gate

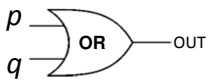
OR Connective vs. **OR Gate**

$p \vee q$

p	q	$p \vee q$
T	T	T
T	F	T
F	T	T
F	F	F



p	q	OUT
1	1	1
1	0	1
0	1	1
0	0	0



"arrowhead block looks like V"

Not Gates

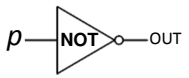
NOT Connective vs. **NOT Gate**

$\neg p$

p	$\neg p$
T	F
F	T

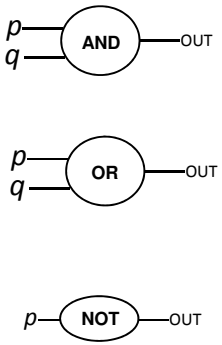
Also called *inverter*

p	OUT
1	0
0	1

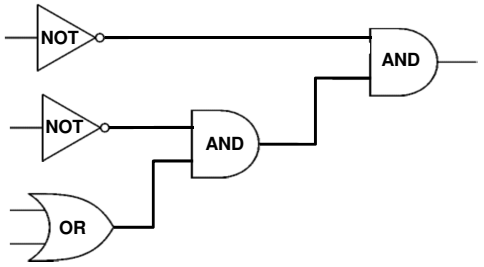


Blobs are Okay!

You may write gates using blobs instead of shapes!

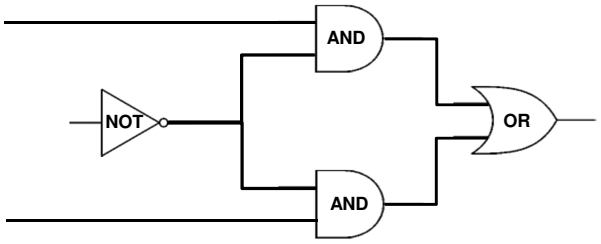


Combinational Logic Circuits



Values get sent along wires connecting gates

Combinational Logic Circuits



Wires can send one value to multiple gates!

Logical Equivalence

Terminology: A compound proposition is a…

- *Tautology* if it is always true
- *Contradiction* if it is always false
- *Contingency* if it can be either true or false

$p \vee \neg p$

$p \oplus p$

$(p \rightarrow q) \wedge p$

$(p \wedge q) \vee (p \wedge \neg q) \vee (\neg p \wedge q) \vee (\neg p \wedge \neg q)$

Logical Equivalence

A and *B* are *logically equivalent* if and only if
A ↔ *B* is a tautology

i.e. A and *B* have the same truth table

The notation *A* ≡ *B* denotes *A* and *B* are logically equivalent

Example: $p \equiv \neg\neg p$

p	¬ *p*	¬ ¬ *p*	*p* ↔ ¬ ¬ *p*

A ↔ *B* vs. *A* ≡ *B*

A ≡ *B* says that ***two*** propositions *A* and *B* *always* ***mean*** *the same thing*

A ↔ *B* is a ***single*** proposition that may be true or false depending on the truth values of the variables in *A* and *B*

- but *A* ≡ *B* and *(A* ↔ *B)* ≡ **T** have the same meaning

Note: Why write *A* ≡ *B* and not *A=B* ?

We use *A=B* to say that *A* and *B* are precisely the same proposition (same sequence of symbols)

De Morgan's laws

$$\neg (p \wedge q) \equiv \neg p \vee \neg q$$
$$\neg (p \vee q) \equiv \neg p \wedge \neg q$$

My code compiles or there is a bug.

De Morgan's laws

Example: $\neg (p \wedge q) \equiv (\neg p \vee \neg q)$

p	*q*	¬ *p*	¬ *q*	¬ *p* ∨ ¬ *q*	*p* ∧ *q*	¬ (*p* ∧ *q*)	¬ (*p* ∧ *q*) ↔ (¬ *p* ∨ ¬ *q*)
T	T						
T	F						
F	T						
F	F						

De Morgan's laws

$$\neg (p \wedge q) \equiv \neg p \vee \neg q$$
$$\neg (p \vee q) \equiv \neg p \wedge \neg q$$

```
if !(front != null && value > front.data)
    front = new ListNode(value, front);
else {
    ListNode current = front;
    while !(current.next == null || current.next.data >= value)
        current = current.next;
    current.next = new ListNode(value, current.next);
}
```

Law of Implication

$$(p \rightarrow q) \equiv (\neg p \vee q)$$

p	q	$p \rightarrow q$	$\neg p$	$\neg p \vee q$	$(p \rightarrow q) \leftrightarrow (\neg p \vee q)$
T	T				
T	F				
F	T				
F	F				

Computing Equivalence

Describe an algorithm for computing if two logical expressions/circuits are equivalent.

What is the run time of the algorithm?

Some Familiar Properties of Arithmetic

- $x + y = y + x$ (Commutativity)
- $x \cdot (y + z) = x \cdot y + x \cdot z$ (Distributivity)
- $(x + y) + z = x + (y + z)$ (Associativity)

Logic has similar properties!

Some Familiar Properties of Arithmetic

- $x + y = y + x$ **(Commutativity)**
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$
- $x \cdot (y + z) = x \cdot y + x \cdot z$ **(Distributivity)**
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- $(x + y) + z = x + (y + z)$ **(Associativity)**
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$

Properties of Logical Connectives

We will always give you this list!

- **Identity**
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- **Domination**
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- **Idempotent**
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- **Commutative**
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$
- **Associative**
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- **Distributive**
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- **Absorption**
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- **Negation**
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

Some Equivalences Related to Implication

$p \rightarrow q \quad \equiv \quad \neg p \vee q$

$p \rightarrow q \quad \equiv \quad \neg q \rightarrow \neg p$

$p \leftrightarrow q \quad \equiv \quad (p \rightarrow q) \wedge (q \rightarrow p)$

$p \leftrightarrow q \quad \equiv \quad \neg p \leftrightarrow \neg q$

Understanding Connectives

- Reflect basic rules of reasoning and logic
- Allow manipulation of logical formulas
  - Simplification
  - Testing for equivalence
- Applications
  - Query optimization
  - Search optimization and caching
  - Artificial Intelligence
  - Program verification