# Boolean Logic

# Applications of Boolean logic

- Computer programs
- And computer addition
- Logic problems
- Sudoku

# Boolean propositions

- A proposition is a statement that can be either true or false
  - “The sky is blue”
  - “I is a Engineering major”
  - “x == y”
- Not propositions:
  - “Are you Bob?”
  - “x := 7”

# Boolean variables

- We use Boolean variables to refer to propositions
  - Usually are lower case letters starting with p (i.e. *p, q, r, s*, etc.)
  - A Boolean variable can have one of two values true (T) or false (F)
- A proposition can be…
  - A single variable: $p$
  - An operation of multiple variables: $p \wedge (q \vee \neg r)$

# Introduction to Logical Operators

- About a dozen logical operators
  - Similar to algebraic operators + * - /
- In the following examples,
  - *p* = "Today is Friday"
  - *q* = "Today is my birthday"

# Logical operators: Not

- A not operation switches (negates) the truth value
- Symbol: ¬ or ~
- In C++ and Java,
  the operand is !

∀ $\neg p$ = "Today is not Friday"

| $p$ | $\neg p$ |
|---|---|
| T | F |
| F | T |

# Logical operators: And

- An and operation is true if both operands are true
- Symbol: $\land$
  - It's like the 'A' in And
- In C++ and Java, the operand is `&&`
- $p \land q$ = "Today is Friday and today is my birthday"

| $p$ | $q$ | $p \land q$ |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | F |
| F | F | F |

# Logical operators: Or

- An or operation is true if either operands are true
- Symbol: $\lor$
- In C++ and Java, the operand is ||
- $p \lor q$ = "Today is Friday or today is my birthday (or possibly both)"

| $p$ | $q$ | $p \lor q$ |
|---|---|---|
| T | T | T |
| T | F | T |
| F | T | T |
| F | F | F |

# Logical operators: Exclusive Or

- An exclusive or operation is true if one of the operands are true, but false if both are true
- Symbol: $\oplus$
- Often called XOR
- $p \oplus q \equiv (p \vee q) \wedge \neg(p \wedge q)$
- In Java, the operand is ^ (but not in C++)
- $p \oplus q$ = "Today is Friday or today is my birthday, but not both"

| $p$ | $q$ | $p \oplus q$ |
|---|---|---|
| T | T | F |
| T | F | T |
| F | T | T |
| F | F | F |

# Inclusive Or versus Exclusive Or

- Do these sentences mean inclusive or exclusive or?
  - Experience with C++ or Java is required
  - Lunch includes soup or salad
  - To enter the country, you need a passport or a driver’s license
  - Publish or perish

# Logical operators: Nand and Nor

- The negation of And and Or, respectively
- Symbols: | and ↓, respectively
  - Nand: $p|q \equiv \neg(p \wedge q)$
  - Nor: $p \downarrow q \equiv \neg(p \vee q)$

| $p$ | $q$ | $p \wedge q$ | $p \vee q$ | $p\|q$ | $p \downarrow q$ |
|---|---|---|---|---|---|
| T | T | T | T | F | F |
| T | F | F | T | T | F |
| F | T | F | T | T | F |
| F | F | F | F | T | T |

# Logical operators: Conditional 1

- A conditional means "if $p$ then $q$"
- Symbol: $\rightarrow$
- $p \rightarrow q$ = "If today is Friday, then today is my birthday"
- $\boxed{p \longrightarrow q \equiv \neg p \vee q}$

**the antecedent**

**the consequence**

| $p$ | $q$ | $p \rightarrow q$ | $\neg p \vee q$ |
|---|---|---|---|
| T | T | T | T |
| T | F | F | F |
| F | T | T | T |
| F | F | T | T |

# Logical operators: Conditional 2

- Let $p$ = "I am elected" and $q$ = "I will lower taxes"
- I state: $p \rightarrow q$ = "If I am elected, then I will lower taxes"
- Consider all possibilities
- Note that if $p$ is false, then the conditional is true regardless of whether $q$ is true or false

| $p$ | $q$ | $p \rightarrow q$ |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | T |
| F | F | T |

# Logical operators: Conditional 3

| | | | | Conditional | Inverse | Converse | Contra-positive |
|---|---|---|---|---|---|---|---|
| $p$ | $q$ | $\neg p$ | $\neg q$ | $p \to q$ | $\neg p \to \neg q$ | $q \to p$ | $\neg q \to \neg p$ |
| T | T | F | F | T | T | T | T |
| T | F | F | T | F | T | T | F |
| F | T | T | F | T | F | F | T |
| F | F | T | T | T | T | T | T |

# Logical operators: Conditional 4

- Alternate ways of stating a conditional:
  - $p$ implies $q$
  - If $p$, $q$
  - $p$ is sufficient for $q$
  - $q$ if $p$
  - $q$ whenever $p$
  - $q$ is necessary for $p$
  - $p$ only if $q$ I don’t like this one

# Logical operators: Bi-conditional 1

- A bi-conditional means "*p* if and only if *q*"
- Symbol: $\leftrightarrow$
- Alternatively, it means "(if *p* then *q*) and (if *q* then *p*)"
- $p \leftrightarrow q \equiv p \rightarrow q \wedge q \rightarrow \mathrm{p}$
- Note that a bi-conditional has the opposite truth values of the exclusive or

| $p$ | $q$ | $p \leftrightarrow q$ |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | F |
| F | F | T |

# Logical operators: Bi-conditional 2

- Let $p$ = “You take this class” and $q$ = “You get a grade”
- Then $p \leftrightarrow q$ means “You take this class if and only if you get a grade”
- Alternatively, it means “If you take this class, then you get a grade and if you get a grade then you take (took) this class”

| $p$ | $q$ | $p \leftrightarrow q$ |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | F |
| F | F | T |

# Boolean operators summary

| | | not | not | and | or | xor | nand | nor | conditional | bi-conditio nal |
|---|---|---|---|---|---|---|---|---|---|---|
| $p$ | $q$ | $\neg p$ | $\neg q$ | $p \wedge q$ | $p \vee q$ | $p \oplus q$ | $p \mid q$ | $p \downarrow q$ | $p \rightarrow q$ | $p \leftrightarrow q$ |
| T | T | F | F | T | T | F | F | F | T | T |
| T | F | F | T | F | T | T | T | F | F | F |
| F | T | T | F | F | T | T | T | F | T | F |
| F | F | T | T | F | F | F | T | T | T | T |

- Learn what they mean, don't just memorize the table!

# Precedence of operators

- Just as in algebra, operators have precedence
  - 4+3*2 = 4+(3*2), not (4+3)*2
- Precedence order (from highest to lowest): $\neg$ $\wedge$ $\vee$ $\rightarrow$ $\leftrightarrow$
  - The first three are the most important
- This means that $p \vee q \wedge \neg r \rightarrow s \leftrightarrow t$ yields: $(p \vee (q \wedge (\neg r))) \leftrightarrow (s \rightarrow t)$
- Not is *always* performed before any other operation

# Translating English Sentences

- Problem:
  - $p$ = "It is below freezing"
  - $q$ = "It is snowing"

- It is below freezing and it is snowing $p \wedge q$
- It is below freezing but not snowing $p \wedge \neg q$
- It is not below freezing and it is not snowing $\neg p \wedge \neg q$
- It is either snowing or below freezing (or both) $p \vee q$
- If it is below freezing, it is also snowing $p \to q$
- It is either below freezing or it is snowing, but it is not snowing if it is below freezing $(p \vee q) \wedge (p \to \neg q)$
- That it is below freezing is necessary and sufficient for it to be snowing $p \leftrightarrow q$

# Translation Example 1

- Heard on the radio:
  - A study showed that there was a correlation between the more children ate dinners with their families and lower rate of substance abuse by those children
  - Announcer conclusions:
    - If children eat more meals with their family, they will have lower substance abuse
    - If they have a higher substance abuse rate, then they did not eat more meals with their family

# Translation Example 1

- Let $p$ = "Child eats more meals with family"
- Let $q$ = "Child has less substance abuse
- Announcer conclusions:
  - If children eat more meals with their family, they will have lower substance abuse
    - $p \rightarrow q$
  - If they have a higher substance abuse rate, then they did not eat more meals with their family
    - $\neg q \rightarrow \neg p$
- Note that $p \rightarrow q$ and $\neg q \rightarrow \neg p$ are logically equivalent

# Translation Example 1

- Let $p$ = "Child eats more meals with family"
- Let $q$ = "Child has less substance abuse"
- Remember that the study showed a *correlation*, not a *causation*

| $p$ | $q$ | result | conclusion |
|---|---|---|---|
| T | T | T | T |
| T | F | ? | F |
| F | T | ? | T |
| F | F | T | T |

# Translation Example 2

- "I have neither given nor received help on this exam"
  - Rephrased: "I have not given nor received …"
  - Let $p$ = "I have given help on this exam"
  - Let $q$ = "I have received help on this exam"
- Translation is: $\neg p \downarrow q$

| $p$ | $q$ | $\neg p$ | $\neg p \downarrow q$ |
|---|---|---|---|
| T | T | F | F |
| T | F | F | T |
| F | T | T | F |
| F | F | T | F |

# Translation Example 2

- What they mean is "I have not given and I have not received help on this exam"
  - Or "I have not (given nor received) help on this exam"

| $p$ | $q$ | $\neg p \wedge \neg q$ | $\neg(p \downarrow q)$ |
|---|---|---|---|
| T | T | F | F |
| T | F | F | F |
| F | T | F | F |
| F | F | T | T |

- The problem: ¬ has a higher precedence than ↓ in Boolean logic, but not always in English
- Also, "neither" is vague

# Boolean Searches

Google™

(101 OR 202) AND bloomfield AND "computer science"

- Note that Google requires you to capitalize Boolean operators
- Google defaults to AND; many others do not
  – So the AND's were optional
- XOR doesn't work…

# Bit Operations 1

- Boolean values can be represented as 1 (true) and 0 (false)
- A bit string is a series of Boolean values
  - 10110100 is eight Boolean values in one string
- We can then do operations on these Boolean strings
  - Each column is its own Boolean operation

$$
\begin{array}{r}
01011010 \\
\oplus\, 10110100 \\
\hline
11101110
\end{array}
$$

# Bit Operations 2

- Evaluate the following

$$11000 \wedge (01011 \vee 11011)$$
$$= 11000 \wedge (11011)$$
$$= 11000$$

$$\begin{array}{r} 01011 \\ \underline{\vee 11011} \\ 11011 \end{array}$$

$$\begin{array}{r} 11000 \\ \underline{\wedge 11011} \\ 11000 \end{array}$$

# && vs. & in C/C++

Consider the following:

```
int p = 11;
int q = 20;
if ( p && q ) {
}
if ( p & q ) {
}
```

In C/C++, any value other than 0 is true

The binary for the integer 11 is 01011

The binary for the integer 20 is 10100

Notice the double ampersand – this is a Boolean operation

As p and q are both true, this is true

Notice the single ampersand – this is a bitwise operation

Bitwise Boolean And operation:

$$\begin{array}{r} 01011 \\ \underline{\wedge 10100} \\ 00000 \end{array}$$

This evaluates to zero (false)!

# && vs. & in C/C++

- Note that Java does not have this “feature”
  - If `p` and `q` are int:
    - `p & q` is bitwise
    - `p && q` will not compile
  - If `p` and `q` are boolean:
    - Both `p & q` and `p && q` will be a Boolean operation
- The same holds true for the or operators (`|` and `||`) in both Java and C/C++

# Tautology and Contradiction

- A tautology is a statement that is always true
  - $p \vee \neg p$ will always be true (Negation Law)
- A contradiction is a statement that is always false
  - $p \wedge \neg p$ will always be false (Negation Law)

| $p$ | $p \vee \neg p$ | $p \wedge \neg p$ |
|---|---|---|
| T | T | F |
| F | T | F |

# Logical Equivalence

- A logical equivalence means that the two sides always have the same truth values
  - Symbol is $\equiv$ or $\Leftrightarrow$
    - We'll use $\equiv$, so as not to confuse it with the bi-conditional

# Logical Equivalences of And

- $p \wedge \mathbf{T} \equiv p$ Identity law

| $p$ | T | $p \wedge$T |
|---|---|---|
| T | T | T |
| F | T | F |

- $p \wedge \mathbf{F} \equiv \mathbf{F}$ Domination law

| $p$ | F | $p \wedge$F |
|---|---|---|
| T | F | F |
| F | F | F |

# Logical Equivalences of And

- $p \wedge p \equiv p$ Idempotent law

| $p$ | $p$ | $p \wedge p$ |
|---|---|---|
| T | T | T |
| F | F | F |

- $p \wedge q \equiv q \wedge p$ Commutative law

| p | q | p∧q | q∧p |
|---|---|---|---|
| T | T | T | T |
| T | F | F | F |
| F | T | F | F |
| F | F | F | F |

# Logical Equivalences of And

- $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$ Associative law

| p | q | r | $p \wedge q$ | $(p \wedge q) \wedge r$ | $q \wedge r$ | $p \wedge (q \wedge r)$ |
|---|---|---|---|---|---|---|
| T | T | T | T | T | T | T |
| T | T | F | T | F | F | F |
| T | F | T | F | F | F | F |
| T | F | F | F | F | F | F |
| F | T | T | F | F | T | F |
| F | T | F | F | F | F | F |
| F | F | T | F | F | F | F |
| F | F | F | F | F | F | F |

# Logical Equivalences of Or

- $p \vee \mathbf{T} \equiv \mathbf{T}$ Identity law
- $p \vee \mathbf{F} \equiv p$ Domination law
- $p \vee p \equiv p$ Idempotent law
- $p \vee q \equiv q \vee p$ Commutative law
- $(p \vee q) \vee r \equiv p \vee (q \vee r)$ Associative law

# Corollary of the Associative Law

- $(p \wedge q) \wedge r \equiv p \wedge q \wedge r$
- $(p \vee q) \vee r \equiv p \vee q \vee r$
- Similar to $(3+4)+5 = 3+4+5$
- Only works if ALL the operators are the same!

# Logical Equivalences of Not

| | |
|---|---|
| • $\neg(\neg p) \equiv p$ | Double negation law |
| • $p \vee \neg p \equiv T$ | Negation law |
| • $p \wedge \neg p \equiv F$ | Negation law |

# DeMorgan's Law

- Probably the most important logical equivalence
- To negate $p \wedge q$ (or $p \vee q$), you "flip" the sign, and negate BOTH p and q
  - Thus, $\neg(p \wedge q) \equiv \neg p \vee \neg q$
  - Thus, $\neg(p \vee q) \equiv \neg p \wedge \neg q$

| p | q | $\neg p$ | $\neg q$ | $p \wedge q$ | $\neg(p \wedge q)$ | $\neg p \vee \neg q$ | $p \vee q$ | $\neg(p \vee q)$ | $\neg p \wedge \neg q$ |
|---|---|---|---|---|---|---|---|---|---|
| T | T | F | F | T | F | F | T | F | F |
| T | F | F | T | F | T | T | T | F | F |
| F | T | T | F | F | T | T | T | F | F |
| F | F | T | T | F | T | T | F | T | T |

# Yet more equivalences

- Distributive:

  $p \lor (q \land r) \equiv (p \lor q) \land (p \lor r)$

  $p \land (q \lor r) \equiv (p \land q) \lor (p \land r)$

- Absorption

  $p \lor (p \land q) \equiv p$

  $p \land (p \lor q) \equiv p$

# How to prove two propositions are equivalent?

- Two methods:
  - Using truth tables
    - Not good for long formulae
    - In this course, only allowed if specifically stated!
  - Using the logical equivalences
    - The preferred method

- Example: show that:

$$(p \to r) \vee (q \to r) \equiv (p \wedge q) \to r$$

# Using Truth Tables

$$(p \to r) \vee (q \to r) \equiv (p \wedge q) \to r$$

| p | q | r | p→r | q →r | (p→r)∨(q →r) | p∧ q | (p∧q) →r |
|---|---|---|---|---|---|---|---|
| T | T | T | T | T | T | T | T |
| T | T | F | F | F | F | T | F |
| T | F | T | T | T | T | F | T |
| T | F | F | F | T | T | F | T |
| F | T | T | T | T | T | F | T |
| F | T | F | T | F | T | F | T |
| F | F | T | T | T | T | F | T |
| F | F | F | T | T | T | F | T |

# Using Logical Equivalences

$$\underline{(p \to r)} \vee \underline{(q \to r)} \equiv \underline{(p \wedge q) \to r}$$

Original statement

Definition of implication

$$p \to q \equiv \neg p \vee q$$

$$(\neg p \vee r) \vee (\neg q \vee r) \equiv \underline{\neg(p \wedge q)} \vee r$$

DeMorgan's Law

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

$$\underline{(\neg p \vee r) \vee (\neg q \vee r)} \equiv (\neg p \vee \neg q) \vee r$$

Associativity of Or

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

$$\neg p \vee r \vee \underline{\neg q} \vee r \equiv \neg p \vee \neg q \vee r$$

Re-arranging

$$\neg p \vee \neg q \vee \underline{r \vee r} \equiv \neg p \vee \neg q \vee r$$

Idempotent Law

$$r \vee r \equiv r$$

$$\neg p \vee \neg q \vee r \equiv \neg p \vee \neg q \vee r$$

# Logical Thinking

- At a trial:
  - Bill says: “Sue is guilty and Fred is innocent.”
  - Sue says: “If Bill is guilty, then so is Fred.”
  - Fred says: “I am innocent, but at least one of the others is guilty.”
- Let b = Bill is innocent, f = Fred is innocent, and s = Sue is innocent
- Statements are:
  - $\neg s \wedge f$
  - $\neg b \rightarrow \neg f$
  - $f \wedge (\neg b \vee \neg s)$

# Can all of their statements be true?

- Show: $(\neg s \wedge f) \wedge (\neg b \rightarrow \neg f) \wedge (f \wedge (\neg b \vee \neg s))$

| b | f | s | ¬b | ¬f | ¬s | ¬s∧f | ¬b⟹¬f | f∧(¬b∨¬s) |
|---|---|---|---|---|---|---|---|---|
| T | T | T | F | F | F | F | T | F |
| T | T | F | F | F | T | T | T | T |
| T | F | T | F | T | F | F | T | F |
| T | F | F | F | T | T | F | T | F |
| F | T | T | T | F | F | F | F | T |
| F | T | F | T | F | T | T | F | T |
| F | F | T | T | T | F | F | T | F |
| F | F | F | T | T | T | F | T | F |

# Are all of their statements true? Show values for s, b, and f such that the equation is true

| | |
|---|---|
| $(\neg s \wedge f) \wedge (\neg b \rightarrow \neg f) \wedge (f \wedge (\neg b \vee \neg s)) \equiv T$ | Original statement |
| $(\neg s \wedge f) \wedge (b \vee \neg f) \wedge (f \wedge (\neg b \vee \neg s)) \equiv T$ | Definition of implication |
| $\neg s \wedge f \wedge (b \vee \neg f) \wedge f \wedge (\neg b \vee \neg s) \equiv T$ | Associativity of AND |
| $\neg s \wedge f \wedge f \wedge (b \vee \neg f) \wedge (\neg b \vee \neg s) \equiv T$ | Re-arranging |
| $\neg s \wedge f \wedge (b \vee \neg f) \wedge (\neg b \vee \neg s) \equiv T$ | Idempotent law |
| $f \wedge (b \vee \neg f) \wedge \neg s \wedge (\neg s \vee \neg b) \equiv T$ | Re-arranging |
| $f \wedge (b \vee \neg f) \wedge \neg s \equiv T$ | Absorption law |
| $(f \wedge (b \vee \neg f)) \wedge \neg s \equiv T$ | Re-arranging |
| $((f \wedge b) \vee (f \wedge \neg f)) \wedge \neg s \equiv T$ | Distributive law |
| $((f \wedge b) \vee F) \wedge \neg s \equiv T$ | Negation law |
| $(f \wedge b) \wedge \neg s \equiv T$ | Domination law |
| $f \wedge b \wedge \neg s \equiv T$ | Associativity of AND |

# What if it weren't possible to assign such values to s, b, and f?

| | |
|---|---|
| $(\neg s \wedge f) \wedge (\neg b \rightarrow \neg f) \wedge (f \wedge (\neg b \vee \neg s)) \wedge s = T$ | Original statement |
| $(\neg s \wedge f) \wedge (b \vee \neg f) \wedge (f \wedge (\neg b \vee \neg s)) \wedge s = T$ | Definition of implication |
| | ... (same as previous slide) |
| $(f \wedge b) \wedge \neg s \wedge s = T$ | Domination law |
| | Re-arranging |
| $f \wedge b \wedge \neg s \wedge s = T$ | |
| $f \wedge b \wedge F = T$ | Negation law |
| $f \wedge F = T$ | Domination law |
| $F = T$ | Domination law |
| | Contradiction! |

# Functional completeness

- All the “extended” operators have equivalences using only the 3 basic operators (and, or, not)
  - The extended operators: nand, nor, xor, conditional, bi-conditional
- Given a limited set of operators, can you write an equivalence of the 3 basic operators?
  - If so, then that group of operators is functionally complete

# Functional completeness of NAND

- Show that | (NAND) is functionally complete

- Equivalence of NOT:
  - $p \mid p \equiv \neg p$
    - $\neg(p \land p) \equiv \neg p$ Equivalence of NAND
    - $\neg(p) \equiv \neg p$ Idempotent law

# Functional completeness of NAND

- Equivalence of AND:
  - $p \wedge q \equiv \neg(p \mid q)$ Definition of nand
  - $p \mid p$ How to do a not using nands
  - $(p \mid q) \mid (p \mid q)$ Negation of $(p \mid q)$
- Equivalence of OR:
  - $p \vee q \equiv \neg(\neg p \wedge \neg q)$ DeMorgan's equivalence of OR
  - As we can do AND and OR with NANDs, we can thus do ORs with NANDs
- Thus, NAND is functionally complete