

22C:19 Discrete Math

Fall 2009
Hantao Zhang

So.... What is it?

- Discrete mathematics ... is the study of mathematical structures that are fundamentally discrete, in the sense of not supporting or requiring the notion of continuity (wikipedia)
- In other words, dealing with integer things (sets, logic, proofs, etc.) and not continuous things (calculus, functions, etc.)

Why discrete math?

- It forms the basis for computer science:
 - Sequences
 - Digital logic (how computers compute)
 - Algorithms
 - Assuring computer correctness
 - Probability and gambling (really!)
 - Etc.
- Like how “regular” math forms the basis for science

Boolean logic

- Oldest and simplest
- Applications:
 - Computer programs
 - Computer addition
 - Logic problems
 - Sudoku

4

Boolean propositions

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

5

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
 - A formula of multiple variables: $p \wedge (q \vee \neg r)$

6

Logical Operators

- About a dozen logical operators
 - Similar to algebraic operators + * - /
- In the following examples,
 - p =: “Today is Friday”
 - q =: “Today is my birthday”
- “=:” reads as “represents” or “means”
 - p is the syntax and “Today is Friday” is the semantics.

7

Logical operators: Not

- A **not** operation switches (negates) the truth value
- Symbol: \neg or \sim
- In C++ and Java, the operand is !
- $\neg p$ =: “Today is not Friday”

p	$\neg p$
T	F
F	T

8

Logical operators: And

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

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

9

Logical operators: Or

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

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

10

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 `^` (in C++, it's bitwise exclusive or)
- $p \oplus q$ =: "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

11

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

12

Logical operators: Nand and Nor

- The negation of And and Or, respectively
- Symbols: $|$ and \downarrow , 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

13

Logical operators: Conditional

- A **conditional** (also called **implication**) means “if p then q ”
- Symbol: \rightarrow
- $p \rightarrow q$ =: “If today is Friday, then today is my birthday”
- $p \rightarrow q \equiv \neg p \vee q$

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

the antecedent the consequence

14

Logical operators: Conditional

- 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

15

Logical operators: Conditional

				Conditional	Inverse	Converse	Contra-positive
p	q	$\neg p$	$\neg q$	$p \rightarrow q$	$\neg p \rightarrow \neg q$	$q \rightarrow p$	$\neg q \rightarrow \neg p$
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

16

Logical operators: Conditional

- 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 ← This one is confusing

17

Logical operators: Bi-conditional

- A bi-conditional means “ p if and only if q ”
- Symbol: \leftrightarrow
- Alternatively, it means “(if p then q) and (if q then p)”
- $p \leftrightarrow q \equiv p \rightarrow q \wedge q \rightarrow p$
- 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

18

Logical operators: Bi-conditional

- 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

19

Boolean operators summary

		not	not	and	or	xor	nand	nor	conditional	bi-conditional
p	q	$\neg p$	$\neg q$	$p \wedge q$	$p \vee q$	$p \oplus q$	$p q$	$p \downarrow q$	$p \rightarrow q$	$p \leftrightarrow q$
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!

20

Precedence of operators

- Just as in algebra, boolean 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 $p \vee q \wedge \neg r \leftrightarrow s \rightarrow t$ yields: $(p \vee (q \wedge (\neg r))) \leftrightarrow (s \rightarrow t)$
- Not** is *always* performed before any other operation

21

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

22

Translation Example

- "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

23

Translation Example

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

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

24

Boolean Searches



(21 OR 22) AND "university of iowa"
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...

25

Bit Operations

- 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

```
01011010
⊕10110100
11101110
```

26

Bit Operations 2

- Evaluate the following

```
11000 ∧ (01011 ∨ 11011)    01011
= 11000 ∧ (11011)          ∨11011
= 11000                      11011
                              11000
                              ∧11011
                              11000
```

27

&& 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 operation: $\begin{array}{r} 01011 \\ \wedge 10100 \\ \hline 00000 \end{array}$ This evaluates to zero (false)!

28

&& 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++
- 29

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

30

What is a Fallacy?

- A fallacy is an error in an argument.
- An argument consists of one or more premises and one conclusion.
- Both the premises (p_1, p_2, \dots, p_n) and the conclusion (q) are statements (i.e., true or false).
- If $(p_1, p_2, \dots, p_n \rightarrow q)$ is not a tautology, then a fallacy occurs.
- If p_i is false, then a fallacy occurs.

31

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

32

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

33

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

34

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

35

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

36

Associative Laws of And and Or

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

37

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

38

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

39

Yet more equivalences

- Distributive:

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

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

- Absorption

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

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

40

How to prove two propositions are equivalent?

- Two methods:

- Using truth tables

- Not good for long formulae

- Using the logical equivalences

- Can be automated

- Example: show that:

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

41

Using Truth Tables

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

p	q	r	$p \rightarrow r$	$q \rightarrow r$	$(p \rightarrow r) \vee (q \rightarrow r)$	$p \wedge q$	$(p \wedge q) \rightarrow r$
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

42

Using Logical Equivalences

$$(p \rightarrow r) \vee (q \rightarrow r) \equiv (p \wedge q) \rightarrow r \quad \text{Original statement}$$

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

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

$$\text{Associativity of } \vee \equiv (\neg p \vee \neg q) \vee r \equiv \neg p \vee \neg q \vee r$$

$$\text{Commutativity of } \vee \equiv \neg p \vee \neg q \vee r \equiv \neg p \vee r \vee \neg q$$

$$\text{Idempotent Law} \equiv \neg p \vee r \equiv \neg p \vee r$$

43

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

44

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	$\neg b$	$\neg f$	$\neg s$	$\neg s \wedge f$	$\neg b \rightarrow \neg f$	$f \wedge (\neg b \vee \neg s)$
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

45

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
 $f \wedge (b \vee \neg f) \wedge \neg s \wedge (\neg b \vee \neg s) \equiv T$ 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

46

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
 $f \wedge b \wedge \neg s \wedge s = T$ Re-arranging
 $f \wedge b \wedge F = T$ Negation law
 $f \wedge F = T$ Domination law
 $F = T$ Domination law
 Contradiction!

47

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

48

Functional completeness of NAND

- Show that $|$ (NAND) is functionally complete
- Equivalence of NOT:
 - $p | p \equiv \neg p$
 - $\neg(p \wedge p) \equiv \neg p$ Equivalence of NAND
 - $\neg(p) \equiv \neg p$ Idempotent law

49

Functional completeness of NAND

- Equivalence of AND:
 - $p \wedge q \equiv \neg(p | q)$ Definition of nand
 - $p | p$ How to do a not using nands
 - $(p | q) | (p | q)$ Negation of $(p | q)$
- 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

50
