1 # CS3203 #1

5/24/04

Janak J Parekh

2 # Intro

- Website location
- Instructor, TA contact info, OH time and location
- Textbook
- Course structure (HW: 6*24 + Q = 90 + F = 90 + class participation)
- Homework structure, submission, lateness
- Exams open-book (midterm 6/14, final 6/30)
- Prerequisite (algebra, basic CS concepts)
- Reasonable Person Principle, lecture material, sleeping
- Cheating, feedback

3 # Motivation

- "Discrete mathematics is the part of mathematics devoted to the study of discrete objects."
- Used when objects are counted, when relationships between finite/countable sets are studied, and processes involving a finite number of steps are analyzed.
  - Examples at the beginning of Rosen
- Serves as a basic mathematics course for many computer science topics
- After all, computers expose a "discrete interface", right?
- It's the algebra you probably never learned in high-school when you moved up to calculus
  - Some duplicates…
- Future classes include set theory, number theory, linear algebra, abstract algebra, combinatorics, graph theory, probability theory

4 # Challenge

- Involves not just simple math, but problem-solving and reasoning skills
- Learning problem solving is a lifelong experience
  - Maybe you'll stump *me* in this class…
- Goes hand-in-hand with becoming not just a good hacker, but a good programmer or a good Computer Scientist

5 # Logic

- First topic, needed to understand the definitions we will use in the rest of the course
- A **proposition** is a declarative sentence that is *either* true *or* false, not both
  - 1+1 = 2
  - Toronto is the capital of Canada

  - What time is it?
  - The sky is blue.
  - x+1 = 2
- We define the **truth value** of a proposition to be *true* (represented as a T) or *false* (represented as a F)
- Leads to **propositional calculus** or **propositional logic**
- Given a proposition, how can we transform it?
  - Given variables, how do we work with them?

6 # Compound propositions and operators

- Formed from simple propositional statements, using **logic operators**
- First operator: **negation** – $\neg p$ suggests "not *p*" or "It is not the case that *p*".
- Negate previous examples?
- We can succintly express this operator using a **truth table**.
- $\neg p$ is a proposition unto itself – negation "produces" a new proposition
  - Unary operator (monadic connective)
  - How many possible unary operators?
  - Example of "counting"
- Binary (dyadic connective) operators

- How many possible such operators?
- We will formally define a few that we find useful

7 Conjunction and disjunction

- "*p* and *q*" = $p \wedge q$ = **conjunction** of *p* and *q*
  - What's the truth table for this?
  - Example: If *p* is "Today is Friday" and *q* is "It is raining today", $p \wedge q$ would produce what?
- "*p* or *q*" is a problem
  - What if *p* and *q* are both true?
  - If the result is true, then **disjunction**, or $p \vee q$
    - "Students who have taken algebra or computer science can take this class."
  - Otherwise, **exclusive or**, i.e., $p \oplus q$
    - "Soup or salad comes with an entrée."
- What about the example from conjunction?
- Let's draw the truth tables…

8 Implication

- Only false if *p* is true and *q* is false
- Represented by $p \rightarrow q$; sometimes called a **conditional statement**
  - "if *p*, then *q*"
  - "*p* implies *q*"
  - **Not** "*p* causes *q*"
- Turns out to be extremely useful
  - "If I am elected, *then* I will lower taxes."
  - "If you get a 100% on the final, then you will get an A."
  - "If it is sunny today, then we will go to the beach."
- Some more interesting examples…
  - "If today is Friday, then 2+3 = 5."
  - "If today is Friday, then 2+3 = 6."
  - "If 2+2 = 5, then you are the Pope."
- Not equivalent to the if-then constructs in programming languages – programming languages short-circuit the concept.

9 Related implications

- Converse: $p \rightarrow q \Rightarrow q \rightarrow p$
- Inverse: $p \rightarrow q \Rightarrow \neg q \rightarrow \neg p$
- These generally are *not* equivalent to the original statement.
  - **Equivalent** suggests both compound propositions have the same truth value.
  - Let's draw the tables…
  - Use previous examples
- However, the **contrapositive** is equivalent to the original implication.
  - Contrapositive: $p \rightarrow q \Rightarrow \neg q \rightarrow \neg p$
  - "If I don't lower taxes, then I am not elected."
  - "If you are not the Pope, then $2+2 \neq 5$."
- You can use these concepts with other binary operators
  - How about disjunction, conjunction, and XOR?

10 Biconditional

- $p \leftrightarrow q$ only holds true if both p and q hold the same truth values
- In other words, "p *if and only if* q"
- Short abbreviation: "p *iff* q"
- Problem: in English, we don't distinguish between implications and biconditionals, and have to use context and guessing.
  - "If you finish your meal, then you can have dessert" generally means biconditional, that is,
    - "If you finish your meal, then you can have dessert" *and*
    - "You can have dessert only if you finish your meal."
  - You could say "If and only if you finish your meal, then you can have dessert", but people generally don't say that
- Note that converse, inverse, and contrapositive of a biconditional hold true

11 Operator precedence

- Negation comes first

- Technically, conjunction takes precedence over disjunction
  - Too difficult to remember, so we use parentheses every time
- Conditional (implication) and biconditional operators have lower precedence
- See table 7, page 10
- Example: $(p \vee \neg q) \rightarrow q$
  - Easiest way to resolve this is to draw a truth table
  - You *might* be able to do this in your head, eventually, but be careful!

12 English ➔ Logic

- Often, need to fill in English ambiguities.
  - "Summers in New York are hot."
  - "If you are in this course, you are not partying tonight, unless you decide to cut out of the second half."
  - "Janak is older than 20 years and younger than 30 years of age."
  - Fortunately, I'm not trying to take advantage of you in this class…
- Specifications
  - "The automated reply cannot be sent when the file system is full."
  - Propositional logic useful for specification
  - **Consistent**: avoid conflicting requirements – must be able to assign truth values to the variables that makes *all* the expressions true
- Mind games
  - "On an island, there are two kinds of inhabitants – knights, who always tell the truth, and the opposte, knaves, who always lie. You encounter two people *A* and *B*. What are they if *A* says "*B* is a knight" and *B* says "the two of us are opposite types"?
  - p = "*A* is a knight" and q = "*B* is a knight" fails

13 Logic and bit operators

- Bits (binary digits) are base-two decimals, used to represent an entity of information in a computer.
- Custom: 1 represents T, 0 represents F
  - Easily translate truth tables to bits
- A variable is a **Boolean variable** if it stores either true or false; we can use a bit as a Boolean variable
- **Bit strings** are a sequence of zero or more bits.
- On a 32-bit machine, 8 bits = 1 byte = 1 character
- We can do AND (conjunction), OR, XOR in a *bitwise* fashion.

14 Propositional equivalences

- A **tautology** is a compound proposition that's always true; one that's always false is a **contradiction**, and all others are **contingencies**.
  - The third term is never used.
- Can you come up with a tautology or a contradiction?
- Another definition of logical equivalence: p and q are logically equivalent if $p \leftrightarrow q$ is a tautology. We call this $p \equiv q$.
- Example: show $\neg(p \vee q) \equiv \neg p \wedge \neg q$
  - DeMorgan's law

15 Well-known equivalences

- See table 5 on page 24
- Identity
- Domination
- Idempotent
- Double negation
- Commutative laws
- Associative laws
- Distributive laws
- DeMorgan's laws
- Absorption laws
- Negation laws
- $p \rightarrow q \equiv \neg p \vee q$

- Implications – contrapositive
- And others…

16 Why use these laws?

- Don't need to generate truth tables for *everything*
- Instead, simplify as much as possible before actually computing a truth table
- Example: show $\neg(p \vee (\neg p \wedge q)) \equiv \neg p \wedge \neg q$ *without* using a truth table

17 Predicates

- Nice to have these propositions, but sometimes we need unspecified (free) variables
  - For example, "x > 3"
- **Predicate** is a declarative statement that becomes a proposition if every free variable is replaced by a constant.
  - "Is greater than 3"
- We define the **propositional function** P(x).
  - For example, P(x): x > 3 holds for P(4), but ¬P(2).
  - "Binding" constants to the free variables
- Can also have multivariable propositional functions.
  - Q(x,y): x = y + 3

18 Quantifiers

- What if we want to generalize for multiple possible values of a variable?
  - We want to **quantify**.
- Two common quantifiers: universal and existential quantifications: leads to **predicate calculus**
  - The *universal quantification* of P(x) is the proposition "P(x) is true for all values of *x* in the universe of discourse"
  - $\forall x P(x)$
  - Note "universe of discourse": very important
  - *Existential quantification:* "There exists an element x in the universe of discourse such that P(x) is true"
  - $\exists x P(x)$
- Example 1
  - P(x): x+1 > x for all real numbers x
  - $\forall x P(x)$ holds true
  - So does $\exists x P(x)$, by definition
- Example 2
  - P(x): $x^2 < 10$ for all positive integers x
  - $\forall x P(x)$ holds false, but $\exists x P(x)$ holds true (1, 2 or 3)

19 Quantifiers, cont'd.

- How to show true or false?
  - For universal, show *one counterexample* x
  - For existential, must show for *all possible counterexamples* x
- Binding
  - As we talked before, can bind a specific value, but quantifiers are *also* a binding
  - Scope of binding: depends on parentheses
  - More complex example: $\forall x(P(x)) \wedge \exists x(Q(x))$
    - If x was the set of integers, give me a P and Q that would produce "true" for this proposition

20 Negation and English

- Much like DeMorgan, we *switch* quantifiers when negating
  - $\forall x \neg P(x) \equiv \neg \exists x(Q(x))$, and vice versa
  - The English equivalent makes sense
- Negation of "There is an honest politician"?
- Need to practice the English equivalents
  - Will be on homework
- Other examples?
  - "Every student in this class has studied calculus."
  - "All lions are carnivorous."
  - Depends on the way you define the universe of discourse.

21 Nested quantifiers

- Don’t need just one quantifier before each expression
  - $\forall x \forall y(x+y=y+x)$ for all real numbers x and y?
  - $\forall x \exists y(x+y=0)$ for all real numbers x and y?
  - Note compact representation of predicate
- Interchanging quantifiers
  - Be careful of interchanging quantifiers of different kinds
  - See table 1, page 50
- Key: understanding what the expression is trying to say
  - Given M(x,y): x is the mother of y where x,y are humans,
  - $\exists x \forall y M(x,y)$: “Someone is a mother to everyone, including herself.”
  - $\forall x \exists y M(x,y)$: “Everyone is a mother to someone.”
  - $\forall y \exists x M(x,y)$: “Everyone has a mother.”
  - $\exists y \forall x M(x,y)$: “Someone has everyone, including herself, as her mother.”

22 Negation with nested quantifiers

- The negation can “propagate through”, and switches each quantifier accordingly
- Example: negate $\forall x \exists y P(x,y)$ such that there is no negation symbol before a quantifier

23 More complicated examples

- Everyone has exactly one best friend.
  - B(x,y): “*y* is the best friend of *x*”
  - $\forall x \exists y(B(x,y) \wedge \forall z((y \neq z) \rightarrow \neg B(x,z)))$
- Every real number, except zero, has a multiplicative inverse.
  - $\forall x((x \neq 0) \rightarrow \exists y(xy = 1))$
  - Same as $\forall x \exists y((x \neq 0) \rightarrow (xy = 1))$?
    - If they evaluate to the same result, then yes
    - Readability
  - Why don’t we have to check for y = 0?

24 Next time

- Proofs, Sets, Functions