IIT KGP
Dept. of Computer Science & Engineering

CS 30053
Foundations of Computing

Debdeep Mukhopadhyay

Mathematical Reasoning

Foundations of Logic

Mathematical Logic is a tool for working with elaborate *compound* statements. It includes:

- A language for expressing them.
- A concise notation for writing them.
- A methodology for objectively reasoning about their truth or falsity.
- It is the foundation for expressing formal proofs in all branches of mathematics.

Foundations of Logic: Overview

1. Propositional logic
2. Predicate logic and Quantifiers
3. Quantifiers and Logical Operators
4. Logical Inference
5. Methods of Proof

Propositional Logic

Propositional Logic is the logic of compound statements built from simpler statements using so-called *Boolean connectives.*

Some applications in computer science:

- Design of digital electronic circuits.
- Expressing conditions in programs.
- Queries to databases & search engines.

George Boole (1815-1864)

Definition of a *Proposition*

Assertion: Statement

Proposition: A *proposition* is an assertion which is either true or false, but not both.

(However, you might not *know* the actual truth value, and it might be situation-dependent.)

[Later in *probability theory* we assign *degrees of certainty* to propositions. But for now: think True/False only!]

Examples of Propositions

- "It is raining." (In a given situation.)
- "Beijing is the capital of China."
- "1 + 2 = 3"

But, the following are **NOT** propositions:

- "Who's there?" (interrogative, question)
- "La la la la la." (meaningless interjection)
- "Just do it!" (imperative, command)

A Paradox

- "I am lying": Is he speaking the truth or lying? True or False??
  - Neither True nor False.
  - If the statement is true, then he says he is lying, that is if he says the truth he is lying
  - If the statement is false, then his statement, "I am lying" is false, which means he is telling the truth
  - Thus, although it appears that the statement is a proposition, this is not. As this cannot be assigned a truth value.

Operators / Connectives

An *operator* or *connective* combines one or more *operand* expressions into a larger expression. (*E.g.*, "+" in numeric exprs.)

Unary operators take 1 operand (*e.g., −*3); *binary* operators take 2 operands (*eg* 3 × 4).

Propositional or *Boolean* operators operate on propositions or truth values instead of on numbers.

Some Popular Boolean Operators

Formal Name	**Nickname**	**Arity**	**Symbol**
Negation operator	NOT	Unary	$\neg$
Conjunction operator	AND	Binary	$\wedge$
Disjunction operator	OR	Binary	$\vee$
Exclusive-OR operator	XOR	Binary	$\oplus$
Implication operator	IMPLIES	Binary	$\Rightarrow$
Biconditional operator	IFF	Binary	$\Leftrightarrow$

The Negation Operator

The unary *negation operator* "¬" (*NOT*) transforms a prop. into its logical *negation*.

E.g. If p = "I have black hair."

then $\neg p$ = "I do **not** have black hair."

Truth table for NOT:

p	$\neg p$
T	F
F	T

Operand column — Result column

T :≡ True; F :≡ False
":≡" means "is defined as"

The Conjunction Operator

The binary *conjunction operator* "$\wedge$" (*AND*) combines two propositions to form their logical *conjunction*.

$\wedge$ND

E.g. If p="I will have salad for lunch." and q="I will have chicken for dinner.", then $p\wedge q$="I will have salad for lunch **and** I will have chicken for dinner."

Remember: "$\wedge$" points up like an "A", and it means "$\wedge$ND"

Conjunction Truth Table

- Note that a conjunction $p_1 \wedge p_2 \wedge \ldots \wedge p_n$ of n propositions will have 2^n rows in its truth table.

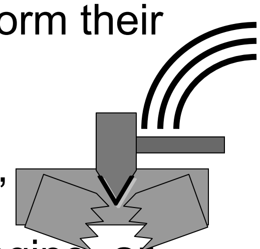
Operand columns

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

- Also: ¬ and ∧ operations together are sufficient to express *any* Boolean truth table!

The Disjunction Operator

The binary *disjunction operator* "∨" (*OR*) combines two propositions to form their logical *disjunction*.



p="My car has a bad engine."

q="My car has a bad carburetor."

$p \vee q$="Either my car has a bad engine, **or** my car has a bad carburetor."

Meaning is like "and/or" in English.

After the downward-pointing "axe" of "∨" splits the wood, you can take 1 piece OR the other, or both.

Disjunction Truth Table

- Note that $p \lor q$ means that p is true, or q is true, **or both** are true!
- So, this operation is also called *inclusive or,* because it **includes** the possibility that both p and q are true.
- "¬" and "∨" together are also universal.

p	q	$p \lor q$	
F	F	F	
F	T	**T**	Note difference from AND
T	F	**T**	
T	T	T	

Nested Propositional Expressions

- Use parentheses to *group sub-expressions*:
  "I just saw my old friend, and either he's grown or I've shrunk." = $f \land (g \lor s)$
  - $(f \land g) \lor s$ would mean something different
  - $f \land g \lor s$ would be ambiguous
- By convention, "¬" takes *precedence* over both "∧" and "∨".
  - $\neg s \land f$ means $(\neg s) \land f$, **not** $\neg (s \land f)$

A Simple Exercise

Let p="It rained last night",
q="The sprinklers came on last night,"
r="The lawn was wet this morning."

Translate each of the following into English:

$\neg p$ = "It didn't rain last night."

$r \wedge \neg p$ = "The lawn was wet this morning, and it didn't rain last night."

$\neg r \vee p \vee q$ = "Either the lawn wasn't wet this morning, or it rained last night, or the sprinklers came on last night."

The *Exclusive Or* Operator

The binary *exclusive-or operator* "$\oplus$" (*XOR*) combines two propositions to form their logical "exclusive or" (exjunction?).

p = "I will earn an A in this course,"

q = "I will drop this course,"

$p \oplus q$ = "I will either earn an A for this course, or I will drop it (but not both!)"

Exclusive-Or Truth Table

- Note that $p \oplus q$ means that p is true, or q is true, but **not both**!
- This operation is called *exclusive or,* because it **excludes** the possibility that both p and q are true.
- "¬" and "⊕" together are **not** universal.

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

Note difference from OR.

Natural Language is Ambiguous

Note that English "or" can be ambiguous regarding the "both" case!

"Pat is a singer or Pat is a writer." - $\vee$

"Pat is a man or Pat is a woman." - $\oplus$

p	q	p "or" q
F	F	F
F	T	T
T	F	T
T	T	?

Need context to disambiguate the meaning!

For this class, assume "or" means inclusive.

The *Implication* Operator

The *implication* $\underbrace{p}_{\text{antecedent}} \Rightarrow \underbrace{q}_{\text{consequent}}$ states that *p* implies *q.*

I.e., If *p* is true, then *q* is true; but if *p* is not true, then *q* could be either true or false.

E.g., let *p* = "You study hard."
q = "You will get a good grade."

$p \Rightarrow q$ = "If you study hard, then you will get a good grade." (else, it could go either way)

Implication Truth Table

- $p \rightarrow q$ is **false** <u>only</u> when *p* is true but *q* is **not** true.
- $p \rightarrow q$ does **not** say that *p* <u>causes</u> *q*!
- $p \rightarrow q$ does **not** require that *p* or *q* <u>**are ever true**</u>!
- *E.g.* "(1=0) → pigs can fly" is TRUE!

p	q	$p \rightarrow q$	
F	F	T	
F	T	T	
T	F	**F**	The <u>only</u> False case!
T	T	T	

For simplicity, I shall denote the implication operator by the symbol → and the iff operator by ↔

Examples of Implications

- "If this lecture ends, then the sun will rise tomorrow." *True* or *False*?
- "If Tuesday is a day of the week, then I am a bird." *True* or *False*?
- "If 1+1=6, then Bush is president." *True* or *False*?
- "If the moon is made of green cheese, then I am richer than Bill Gates." *True* or *False*?

Why does this seem wrong?

- Consider a sentence like,
  - "If I wear a red shirt tomorrow, then Arnold Schwarzenegger will become governor of California."
- In logic, we consider the sentence **True** so long as either I don't wear a red shirt, or Arnold wins.
- But in normal English conversation, if I were to make this claim, you would think I was lying.
  - Why this discrepancy between logic & language?

Resolving the Discrepancy

- In English, a sentence "if *p* then *q*" usually really *implicitly* means something like,
  - "In all possible situations, if *p* then *q*."
    - That is, "For *p* to be true and *q* false is *impossible*."
    - Or, "I *guarantee* that no matter what, if *p*, then *q*."
- This can be expressed in *predicate logic* as:
  - "For all situations *s*, if *p* is true in situation *s*, then *q* is also true in situation *s*"
  - Formally, we could write: $\forall s, P(s) \rightarrow Q(s)$
- *That* sentence is logically ***False*** in our example, because for me to wear a red shirt and for Arnold to lose is a *possible* (even if not actual) situation.
  - Natural language and logic then agree with each other.

English Phrases Meaning $p \rightarrow q$

- "*p* implies *q*"
- "if *p*, then *q*"
- **"if *p*, *q*"**
- "when *p*, *q*"
- "whenever *p*, *q*"
- "*q* if *p*"
- "*q* when *p*"
- "*q* whenever *p*"
- **"*p* only if *q*"**
- **"*p* is sufficient for *q*"**
- **"*q* is necessary for *p*"**
- "*q* follows from *p*"
- "*q* is implied by *p*"

If p is true, that is enough, **q has to be true** for the implication to hold (sufficiency)
If q is false, p cannot be true; It is necessary that q be true for p to be true (necessicity)

Converse, Inverse, Contrapositive

Some terminology, for an implication $p \rightarrow q$:

- Its *converse* is: $q \rightarrow p$.
- Its *inverse* is: $\neg p \rightarrow \neg q$.
- Its *contrapositive*: $\neg q \rightarrow \neg\, p$.
- One of these three has the *same meaning* (same truth table) as $p \rightarrow q$. Can you figure out which?

Contrapositive

How do we know for sure?

Proving the equivalence of $p \rightarrow q$ and its contrapositive using truth tables:

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

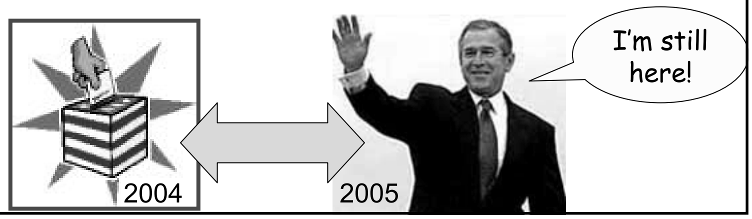
The *biconditional* operator

The *biconditional* $p \leftrightarrow q$ states that p is true *if and only if (IFF)* q is true.

p = "Bush wins the 2004 election."

q = "Bush will be president for all of 2005."

$p \leftrightarrow q$ = "If, and only if, Bush wins the 2004 election, Bush will be president for all of 2005."



Biconditional Truth Table

- $p \leftrightarrow q$ means that p and q have the **same** truth value.
- Note this truth table is the exact **opposite** of $\oplus$'s!
  Thus, $p \leftrightarrow q$ means $\neg(p \oplus q)$
- $p \leftrightarrow q$ does **not** imply that p and q are true, or cause each other.

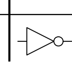
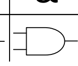
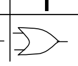
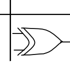
p	q	$p \leftrightarrow q$
F	F	T
F	T	F
T	F	F
T	T	T

Boolean Operations Summary

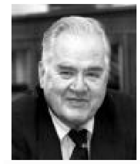
- We have seen 1 unary operator (out of the 4 possible) and 5 binary operators (out of the 16 possible). Their truth tables are below.

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \oplus q$	$p \rightarrow q$	$p \leftrightarrow q$
F	F	T	F	F	F	T	T
F	T	T	F	T	T	T	F
T	F	F	F	T	T	F	F
T	T	F	T	T	F	T	T

Some Alternative Notations

Name:	not	and	or	xor	implies	iff
Propositional logic:	$\neg$	$\wedge$	$\vee$	$\oplus$	$\rightarrow$	$\leftrightarrow$
Boolean algebra:	$\overline{p}$	pq	$+$	$\oplus$		
C/C++/Java (wordwise):	!	&&	||	!=		==
C/C++/Java (bitwise):	~	&	|	^		
Logic gates:						

Bits and Bit Operations



John Tukey (1915-2000)

- A *bit* is a binary (base 2) digit: 0 or 1.
- Bits may be used to represent truth values.
- By convention:
  0 represents "false"; 1 represents "true".
- *Boolean algebra* is like ordinary algebra except that variables stand for bits, + means "or", and multiplication means "and".

Bit Strings

- A *Bit string* of *length n* is an ordered series or sequence of $n \geq 0$ bits.
  - More on sequences in §3.2.
- By convention, bit strings are written left to right: *e.g.* the first bit of "1001101010" is 1.
- When a bit string represents a base-2 number, by convention the first bit is the *most significant* bit. *Ex.* $1101_2=8+4+1=13$.

Counting in Binary

- Did you know that you can count to 1,023 just using two hands?
  - How? Count in binary!
    - Each finger (up/down) represents 1 bit.
- To increment: Flip the rightmost (low-order) bit.
  - If it changes 1→0, then also flip the next bit to the left,
    - If that bit changes 1→0, then flip the next one, *etc.*
- 0000000000, 0000000001, 0000000010, …
  …, 1111111101, 1111111110, 1111111111

Bitwise Operations

- Boolean operations can be extended to operate on bit strings as well as single bits.
- E.g.:
  01 1011 0110
  <u>11 0001 1101</u>
  11 1011 1111 Bit-wise OR
  01 0001 0100 Bit-wise AND
  10 1010 1011 Bit-wise XOR

Summary

You have learned about:

- Propositions: What they are.
- Propositional logic operators'
  - Symbolic notations.
  - English equivalents.
  - Logical meaning.
  - Truth tables.
- Nested propositions.
- Alternative notations.
- Bits and bit-strings.
- Next section:
  - Propositional equivalences.
  - How to prove them.

Propositional Equivalence

Two *syntactically* (*i.e.,* textually) different compound propositions may be the *semantically* identical (*i.e.,* have the same meaning). We call them *equivalent*. Learn:

- Various *equivalence rules* or *laws*.
- How to *prove* equivalences using *symbolic derivations*.

Tautologies and Contradictions

A *tautology* is a compound proposition that is **true** *no matter what* the truth values of its atomic propositions are!

Ex. $p \vee \neg p$ [What is its truth table?]

A *contradiction* is a compound proposition that is **false** no matter what! *Ex.* $p \wedge \neg p$ [Truth table?]

Other compound props. are *contingencies*
(which is neither a tautology nor a contradiction)

Logical Equivalence

Compound proposition p is *logically equivalent* to compound proposition q, written $p \Leftrightarrow q$, **IFF** the compound proposition $p \leftrightarrow q$ is a tautology.

Compound propositions p and q are logically equivalent to each other **IFF** p and q contain the same truth values as each other in <u>all</u> rows of their truth tables.

Proving Equivalence via Truth Tables

Ex. Prove that $p \vee q \Leftrightarrow \neg(\neg p \wedge \neg q)$.

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

Constructing Truth table

Construct a truth table for $q \wedge \neg p \rightarrow p$.

p	q	$\neg p$	$\neg p \wedge q$	$q \wedge \neg p \rightarrow p$
F	F	T	F	T
F	T	T	T	F
T	F	F	F	T
T	T	F	F	T

Equivalence Laws

- These are similar to the arithmetic identities you may have learned in algebra, but for propositional equivalences instead.
- They provide a pattern or template that can be used to match all or part of a much more complicated proposition and to find an equivalence for it.

Equivalence Laws - Examples

- *Identity*: $p \wedge \mathbf{T} \Leftrightarrow p \quad p \vee \mathbf{F} \Leftrightarrow p$
- *Domination*: $p \vee \mathbf{T} \Leftrightarrow \mathbf{T} \quad p \wedge \mathbf{F} \Leftrightarrow \mathbf{F}$
- *Idempotent*: $p \vee p \Leftrightarrow p \quad p \wedge p \Leftrightarrow p$
- *Double negation:* $\neg\neg p \Leftrightarrow p$
- *Commutative:* $p \vee q \Leftrightarrow q \vee p \quad p \wedge q \Leftrightarrow q \wedge p$
- *Associative:* $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$
  $(p \wedge q) \wedge r \Leftrightarrow p \wedge (q \wedge r)$

More Equivalence Laws

- *Distributive*: $p\vee(q\wedge r) \Leftrightarrow (p\vee q)\wedge(p\vee r)$
  $p\wedge(q\vee r) \Leftrightarrow (p\wedge q)\vee(p\wedge r)$
- *De Morgan's*:
  $\neg(p\wedge q) \Leftrightarrow \neg p \vee \neg q$
  $\neg(p\vee q) \Leftrightarrow \neg p \wedge \neg q$
- *Trivial tautology/contradiction*:
  $p \vee \neg p \Leftrightarrow \mathbf{T}$ $\quad p \wedge \neg p \Leftrightarrow \mathbf{F}$

Augustus De Morgan (1806-1871)

Defining Operators via Equivalences

Using equivalences, we can *define* operators in terms of other operators.

- Exclusive or: $p\oplus q \Leftrightarrow (p\vee q)\wedge\neg(p\wedge q)$
  $p\oplus q \Leftrightarrow (p\wedge\neg q)\vee(q\wedge\neg p)$
- Implies: $p\rightarrow q \Leftrightarrow \neg p \vee q$
- Biconditional: $p\leftrightarrow q \Leftrightarrow (p\rightarrow q) \wedge (q\rightarrow p)$
  $p\leftrightarrow q \Leftrightarrow \neg(p\oplus q)$

Topic #1.1 – Propositional Logic: Equivalences

An Example Problem

- Check using a symbolic derivation whether $(p \wedge \neg q) \rightarrow (p \oplus r) \Leftrightarrow \neg p \vee q \vee \neg r$.

$(p \wedge \neg q) \underline{\rightarrow} (p \oplus r) \Leftrightarrow$

[Expand definition of $\rightarrow$] $\neg(p \wedge \neg q) \vee (p \underline{\oplus} r)$

[Defn. of $\oplus$] $\Leftrightarrow \neg(p \wedge \neg q) \vee ((p \vee r) \wedge \neg(p \wedge r))$

[DeMorgan's Law]

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

cont.

Topic #1.1 – Propositional Logic: Equivalences

Example Continued...

$(\neg p \vee q) \vee ((p \vee r) \wedge \neg(p \wedge r)) \Leftrightarrow$ [$\vee$ commutes]

$\Leftrightarrow \underline{(q \vee \neg p)} \vee ((p \vee r) \wedge \neg(p \wedge r))$ [$\vee$ associative]

$\Leftrightarrow q \vee (\underline{\neg p \vee} ((p \vee r) \wedge \neg(p \wedge r)))$ [distrib. $\vee$ over $\wedge$]

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

[assoc.] $\Leftrightarrow q \vee ((\underline{(\neg p \vee p) \vee r}) \wedge (\neg p \vee \neg(p \wedge r)))$

[trivail taut.] $\Leftrightarrow q \vee ((\underline{\mathbf{T}} \vee r) \wedge (\neg p \vee \neg(p \wedge r)))$

[domination] $\Leftrightarrow q \vee (\underline{\mathbf{T}} \wedge (\neg p \vee \neg(p \wedge r)))$

[identity] $\Leftrightarrow q \vee (\neg p \vee \neg(p \wedge r)) \Leftrightarrow$ *cont.*

End of Long Example

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

[DeMorgan's] $\Leftrightarrow q \vee (\neg p \vee (\neg p \vee \neg r))$

[Assoc.] $\Leftrightarrow q \vee ((\neg p \vee \neg p) \vee \neg r)$

[Idempotent] $\Leftrightarrow q \vee (\neg p \vee \neg r)$

[Assoc.] $\Leftrightarrow (q \vee \neg p) \vee \neg r$

[Commut.] $\Leftrightarrow \neg p \vee q \vee \neg r$

Q.E.D. (quod erat demonstrandum)

(Which was to be shown.)

Review: Propositional Logic

- Atomic propositions: p, q, r, …
- Boolean operators: $\neg\ \wedge\ \vee\ \oplus\ \rightarrow\ \leftrightarrow$
- Compound propositions: $s :\equiv (p \wedge \neg q) \vee r$
- Equivalences: $p \wedge \neg q \Leftrightarrow \neg(p \rightarrow q)$
- Proving equivalences using:
  - Truth tables.
  - Symbolic derivations. $p \Leftrightarrow q \Leftrightarrow r$ …

Predicate Logic

- Language of propositions not sufficient to make all assertions needed in mathematics
  - $x=3$, $x+y=z$
  - They are not propositions (Why?)
  - However if values are assigned they do
- Consider the assertion:
  - He is tall and dark
  - These assertions are formed using variables, in a template. The template is called the predicate.

Contd…

- Assertion : x is tall and dark.
  - x is the variable
  - “is tall and dark” is the predicate

Applications of Predicate Logic

It is *the* formal notation for writing perfectly clear, concise, and unambiguous mathematical *definitions*, *axioms*, and *theorems* for *any* branch of mathematics.

Predicate logic with function symbols, the “=” operator, and a few proof-building rules is sufficient for defining *any* conceivable mathematical system, and for proving anything that can be proved within that system!

Other Applications

Kurt Gödel
1906-1978

- Predicate logic is the foundation of the field of *mathematical logic*, which culminated in *Gödel’s incompleteness theorem*, which revealed the ultimate limits of mathematical thought:
  - Given any finitely describable, consistent proof procedure, there will still be *some* true statements that can *never be proven* by that procedure.
- *I.e.*, we can’t discover *all* mathematical truths, unless we sometimes resort to making *guesses.*

Practical Applications

- Basis for clearly expressed formal specifications for any complex system.
- Basis for *automatic theorem provers* and many other Artificial Intelligence systems.

Subjects and Predicates

- In the sentence “The dog is sleeping”:
  - The phrase “the dog” denotes the *subject* - the *object* or *entity* that the sentence is about.
  - The phrase “is sleeping” denotes the *predicate*- a property that is true **of** the subject.
- In predicate logic, a *predicate* is modeled as a *function* $P(\cdot)$ from objects to propositions.
  - $P(x)$ = “x is sleeping” (where x is any object).

More About Predicates

- Convention: Lowercase variables *x*, *y*, *z*... denote objects/entities; uppercase variables *P*, *Q*, *R*… denote propositional functions (predicates).
- Keep in mind that the *result of applying* a predicate *P* to an object *x* is the *proposition* $P(x)$. But the predicate *P* **itself** (*e.g. P*="is sleeping") is **not** a proposition (not a complete sentence).
  - *E.g.* if $P(x)$ = "*x* is a prime number", $P(3)$ is the *proposition* "3 is a prime number."

Propositional Functions

- Predicate logic *generalizes* the grammatical notion of a predicate to also include propositional functions of **any** number of arguments, each of which may take **any** grammatical role that a noun can take.
  - *E.g.* let $P(x,y,z)$ = "*x* gave *y* the grade *z*", then if *x*="Mike", *y*="Mary", *z*="A", then $P(x,y,z)$ = "Mike gave Mary the grade A."

Universes of Discourse (U.D.s)

- The power of distinguishing objects from predicates is that it lets you state things about *many* objects at once.
- E.g., let $P(x)$="x+1>x". We can then say,
  "For *any* number x, $P(x)$ is true" instead of
  $(\mathbf{0}+1>\mathbf{0}) \wedge (\mathbf{1}+1>\mathbf{1}) \wedge (\mathbf{2}+1>\mathbf{2}) \wedge \ldots$
- The collection of values that a variable x can take is called x's *universe of discourse*.

Types of predicates

- Consider a predicate: P(c1,c2,…,cn)
- **Defn:**
  - **Valid:** Value of P is true for all choices of the argument
  - **Satisfiable:** Value of P is true for some value of the argument
  - **Unsatisfiable:** Value of P is never true for the possible choices of the argument

Quantifier Expressions

- *Quantifiers* provide a notation that allows us to *quantify* (count) *how many* objects in the univ. of disc. satisfy a given predicate.
- "$\forall$" is the FOR$\forall$LL or *universal* quantifier. $\forall x\ P(x)$ means *for all* x in the u.d., P holds.
- "$\exists$" is the $\exists$XISTS or *existential* quantifier. $\exists x\ P(x)$ means <u>there *exists*</u> an x in the u.d. (that is, 1 or more) <u>such that</u> $P(x)$ is true.

The Universal Quantifier $\forall$

- Example:
  Let the u.d. of x be <u>parking spaces at IITKGP</u>.
  Let $P(x)$ be the *predicate* "x is full."
  Then the *universal quantification of* $P(x)$, $\forall x\ P(x)$, is the *proposition:*
  - "All parking spaces at IITKGP are full."
  - *i.e.*, "Every parking space at IITKGP is full."
  - *i.e.*, "For each parking space at IITKGP, that space is full."

The Existential Quantifier ∃

- Example:
  Let the u.d. of x be parking spaces at IITKGP.
  Let *P*(*x*) be the *predicate* "*x* is full."
  Then the *existential quantification of P*(*x*), ∃*x P*(*x*), is the *proposition*:
  - "Some parking space at IITKGP is full."
  - "There is a parking space at IITKGP that is full."
  - "At least one parking space at IITKGP is full."

Question

- What is a predicate with zero variables called?

Free and Bound Variables

- An expression like $P(x)$ is said to have a *free variable* x (meaning, x is undefined).
- A quantifier (either $\forall$ or $\exists$) *operates* on an expression having one or more free variables, and *binds* one or more of those variables, to produce an expression having one or more *bound variables*.
- Binding converts a predicate to a proposition

Example of Binding

- $P(x,y)$ has 2 free variables, x and y.
- $\forall x\, P(x,y)$ has 1 free variable, and one bound variable. [Which is which?]
- "$P(x)$, where x=3" is another way to bind x.
- An expression with <u>zero</u> free variables is a bona-fide (actual) proposition
- An expression with <u>one or more</u> free variables is still only a predicate: $\forall x\, P(x,y)$

Nesting of Quantifiers

Example: Let the u.d. of x & y be people.

Let $L(x,y)$="x likes y" (a predicate w. 2 f.v.'s)

Then $\exists y\, L(x,y)$ = "There is someone whom x likes." (A predicate w. 1 free variable, x)

Then $\forall x\, (\exists y\, L(x,y))$ =
"Everyone has someone whom they like."
(A __Proposition__ with __0__ free variables.)

Review: Propositional Logic

- Atomic propositions: p, q, r, …
- Boolean operators: $\neg\ \wedge\ \vee\ \oplus\ \rightarrow\ \leftrightarrow$
- Compound propositions: $s \equiv (p \wedge \neg q) \vee r$
- Equivalences: $p\wedge\neg q \Leftrightarrow \neg(p \rightarrow q)$
- Proving equivalences using:
  - Truth tables.
  - Symbolic derivations. $p \Leftrightarrow q \Leftrightarrow r$ …

Review: Predicate Logic

- Objects *x*, *y*, *z*, …
- Predicates *P*, *Q*, *R*, … are functions mapping objects *x* to propositions *P*(*x*).
- Multi-argument predicates *P*(*x*, *y*).
- Quantifiers: [$\forall x\ P(x)$] :≡ "For all *x*'s, *P*(*x*)." [$\exists x\ P(x)$] :≡ "There is an *x* such that *P*(*x*)."
- Universes of discourse, bound & free vars.

Quantifier Exercise

If *R*(*x*,*y*)="*x* relies upon *y*," express the following in unambiguous English:

$\forall x(\exists y\ R(x,y))$= Everyone has *someone* to rely on.

$\exists y(\forall x\ R(x,y))$= There's a poor overburdened soul whom *everyone* relies upon (including himself)!

$\exists x(\forall y\ R(x,y))$= There's some needy person who relies upon *everybody* (including himself).

$\forall y(\exists x\ R(x,y))$= Everyone has *someone* who relies upon them.

$\forall x(\forall y\ R(x,y))$= *Everyone* relies upon *everybody*, (including themselves)!

Natural language is ambiguous!

- "Everybody likes somebody."
  - For everybody, there is somebody they like,
    - $\forall x\ \exists y\ Likes(x,y)$ [Probably more likely.]
  - or, there is somebody (a popular person) whom everyone likes?
    - $\exists y\ \forall x\ Likes(x,y)$
- "Somebody likes everybody."
  - Same problem: Depends on context, emphasis.

Game Theoretic Semantics

- Thinking in terms of a competitive game can help you tell whether a proposition with nested quantifiers is true.
- The game has two players, <u>both with the same knowledge</u>:
  - Verifier: Wants to demonstrate that the proposition is true.
  - Falsifier: Wants to demonstrate that the proposition is false.
- The Rules of the Game "Verify or Falsify":
  - Read the quantifiers from <u>left to right</u>, picking values of variables.
  - When you see "∀", the falsifier gets to select the value.
  - When you see "∃", the verifier gets to select the value.
- If the verifier <u>can always win</u>, then the proposition is true.
- If the falsifier <u>can always win</u>, then it is false.

Let's Play, "Verify or Falsify!"

Let $B(x,y) :\equiv$ "x's month of birthday is the same as that of y"

Suppose I claim that among you:

$\underline{\forall x}\ \exists y\ B(x,y)$

Your turn, as falsifier:
You pick any $x \rightarrow$ *(so-and-so)*

$\underline{\exists y}\ B(\text{so-and-so},y)$

My turn, as verifier:
I pick any $y \rightarrow$ *(such-and-such)*

$B(\text{so-and-so},\text{such-and-such})$

- Let's play it in class.
- Who wins this game?
- What if I switched the quantifiers, and I claimed that $\exists y\ \forall x\ B(x,y)$? Who wins in that case?

Still More Conventions

- Sometimes the universe of discourse is restricted within the quantification, *e.g.*,
  - $\forall x{>}0\ P(x)$ is shorthand for "For all x that are greater than zero, $P(x)$."
    $=\forall x\ (x{>}0 \rightarrow P(x))$
  - $\exists x{>}0\ P(x)$ is shorthand for "There is an x greater than zero such that $P(x)$."
    $=\exists x\ (x{>}0 \wedge P(x))$

More to Know About Binding

- $\forall x\ \exists x\ P(x)$ - x is not a free variable in $\exists x\ P(x)$, therefore the $\forall x$ binding isn't used.
- $(\forall x\ P(x)) \wedge Q(x)$ - The variable x is outside of the *scope* of the $\forall x$ quantifier, and is therefore free. Not a complete proposition!
- $(\forall x\ P(x)) \wedge (\exists x\ Q(x))$ – This is legal, because there are 2 different x's!

Commutavity of Quantifiers

- $\forall x\ \exists y\ P(x,y) \neq \exists y \forall x\ P(x,y)$
- $\forall x\ \forall y\ P(x,y) = \forall y \forall x\ P(x,y)$
- $\exists\ x\ \exists y\ P(x,y) = \exists y \exists x P(x,y)$

It is easy to disprove (give a counter-example)
Prove or disprove the above statements

Quantifier Equivalence Laws

- Definitions of quantifiers: If u.d.=a,b,c,…
  $\forall x\, P(x) \Leftrightarrow P(\mathrm{a}) \wedge P(\mathrm{b}) \wedge P(\mathrm{c}) \wedge \ldots$
  $\exists x\, P(x) \Leftrightarrow P(\mathrm{a}) \vee P(\mathrm{b}) \vee P(\mathrm{c}) \vee \ldots$
- From those, we can prove the laws:
  $\forall x\, P(x) \Leftrightarrow \neg\exists x\, \neg P(x)$
  $\exists x\, P(x) \Leftrightarrow \neg\forall x\, \neg P(x)$
  DeMorgan's
- Which *propositional* equivalence laws can be used to prove this?

More Equivalence Laws

- $\forall x\, \forall y\, P(x,y) \Leftrightarrow \forall y\, \forall x\, P(x,y)$
  $\exists x\, \exists y\, P(x,y) \Leftrightarrow \exists y\, \exists x\, P(x,y)$
- $\forall x\, \mathbf{(}P(x) \wedge Q(x)\mathbf{)} \Leftrightarrow \mathbf{(}\forall x\, P(x)\mathbf{)} \wedge \mathbf{(}\forall x\, Q(x)\mathbf{)}$
  $\exists x\, \mathbf{(}P(x) \vee Q(x)\mathbf{)} \Leftrightarrow \mathbf{(}\exists x\, P(x)\mathbf{)} \vee \mathbf{(}\exists x\, Q(x)\mathbf{)}$
- Exercise:
  See if you can prove these yourself.
  - What propositional equivalences did you use?

Review: Predicate Logic

- Objects x, y, z, …
- Predicates P, Q, R, … are functions mapping objects x to propositions $P(x)$.
- Multi-argument predicates $P(x, y)$.
- Quantifiers: $(\forall x\, P(x))$ =“For all x’s, $P(x)$.” $(\exists x\, P(x))$=“There is an x such that $P(x)$.”

Defining New Quantifiers

As per their name, quantifiers can be used to express that a predicate is true of any given *quantity* (number) of objects.

Define $\exists! x\, P(x)$ to mean “$P(x)$ is true of *exactly one* x in the universe of discourse.”

$\exists! x\, P(x) \Leftrightarrow \exists x\, (P(x) \wedge \neg\exists y\, (P(y) \wedge y \neq x))$ “There is an x such that $P(x)$, where there is no y such that P(y) and y is other than x.”

More about Quantifiers

- State True or False with reasons:
  - $\forall$ distributes over $\wedge$
  - $\forall$ distributes over $\vee$
  - $\exists$ distributes over $\wedge$
  - $\exists$ distributes over $\vee$
  - $\exists x[P(x) \wedge Q(x)] \rightarrow \exists xP(x) \wedge \exists xQ(x)$
  - $\forall x[P(x) \vee Q(x)] \rightarrow \forall xP(x) \vee \forall xQ(x)$

Prove or disprove: $\exists x[P(x) \rightarrow Q(x)] \Leftrightarrow [\exists xP(x) \rightarrow \exists xQ(x)]$

$\exists x[P(x) \rightarrow Q(x)] \Leftrightarrow \exists x[\neg P(x) \vee Q(x)]$

$\Leftrightarrow \exists x[\neg P(x)] \vee \exists xQ(x) \Leftrightarrow \neg \forall xP(x) \vee \exists xQ(x)$

$\Leftrightarrow \forall xP(x) \rightarrow \exists xQ(x)$

Hence we are to check:

$[\forall xP(x) \rightarrow \exists xQ(x)] \Leftrightarrow [\exists xP(x) \rightarrow \exists xQ(x)]$

Truth Table

$\forall xP(x)$	$\exists xP(x)$	$\exists xQ(x)$	$\forall xP(x) \rightarrow \exists xQ(x)$	$\exists xP(x) \rightarrow \exists xQ(x)$
0	0	0	1	1
0	0	1	1	1
0	1	0	1	0
0	1	1	1	1
1	0	0	n.a	n.a
1	0	1	n.a	n.a
1	1	0	0	0
1	1	1	1	1

Building Counter-example

- Build the counter-example, so that we satisfy the line of the truth-table which makes the difference:
  - Here, $\forall xP(x)=0$, $\exists xP(x)=1$, $\exists xQ(x)=0$
  - Example: P(x) is satisfiable and Q(x) is unsatisfiable
  - P(x): x=0, Q(x): $x \neq x$.

Some Number Theory Examples

- Let u.d. = the *natural numbers* 0, 1, 2, …
- "A number x is *even*, $E(x)$, if and only if it is equal to 2 times some other number."
  $\forall x\ (E(x) \leftrightarrow (\exists y\ \ x=2y))$
- "A number is *prime*, $P(x)$, iff it's greater than 1 and it isn't the product of two non-unity numbers."
  $\forall x\ (P(x) \leftrightarrow (x>1 \wedge \neg\exists yz\ \ x=yz \wedge y\neq 1 \wedge z\neq 1))$

Goldbach's Conjecture (unproven)

Using $E(x)$ and $P(x)$ from previous slide,

$\forall E(x>2)$: $\exists P(p),P(q)$: $p+q = x$

or, with more explicit notation:

$$\forall x\ [x>2 \wedge E(x)] \rightarrow$$
$$\exists p\ \exists q\ P(p) \wedge P(q) \wedge p+q = x.$$

"Every even number greater than 2 is the sum of two primes."

Deduction Example

- Definitions:
  - s :≡ Socrates (ancient Greek philosopher);
  - $H(x)$:≡ "x is human";
  - $M(x)$:≡ "x is mortal".
- Premises:
  - $H(\mathrm{s})$ *Socrates is human.*
  - $\forall x\, H(x) \rightarrow M(x)$ *All humans are mortal.*

Prove, Socrates is mortal!!

Deduction Example Continued

Some valid conclusions you can draw:

$H(\mathrm{s}) \rightarrow M(\mathrm{s})$ **[Instantiate universal.]** *If Socrates is human then he is mortal.*

$\neg H(\mathrm{s}) \vee M(\mathrm{s})$ *Socrates is inhuman or mortal.*

$H(\mathrm{s}) \wedge (\neg H(\mathrm{s}) \vee M(\mathrm{s}))$
Socrates is human, and also either inhuman or mortal.

$(H(\mathrm{s}) \wedge \neg H(\mathrm{s})) \vee (H(\mathrm{s}) \wedge M(\mathrm{s}))$ **[Apply distributive law.]**

$\mathbf{F} \vee (H(\mathrm{s}) \wedge M(\mathrm{s}))$ **[Trivial contradiction.]**

$H(\mathrm{s}) \wedge M(\mathrm{s})$ **[Use identity law.]**

$M(\mathrm{s})$ *Socrates is mortal.*

Another Example

- Definitions: $H(x) :\equiv$ "x is human"; $M(x) :\equiv$ "x is mortal"; $G(x) :\equiv$ "x is a god"
- Premises:
  - $\forall x\ H(x) \rightarrow M(x)$ ("Humans are mortal") and
  - $\forall x\ G(x) \rightarrow \neg M(x)$ ("Gods are immortal").
- Show that $\neg\exists x\ (H(x) \wedge G(x))$
  ("No human is a god.")

Summary

- From these sections you should have learned:
  - Predicate logic notation & conventions
  - Conversions: predicate logic $\leftrightarrow$ clear English
  - Meaning of quantifiers, equivalences
  - Simple reasoning with quantifiers
- Upcoming topics:
  - Introduction to proof-writing.
  - Then: Set theory –
    - a language for talking about collections of objects.