VCU, Department of Computer Science

CMSC 302

Propositional Logic

Vojislav Kecman

30-Jan-17

All the PPT slides are based on MP Franck's and H Bingöl's ones

The Fundamentals of Logic

In Rosen, §§1.1-1.4

~98 slides, ~3 lectures

30-Jan-17

Foundations of Logic

30-Jan-17

Foundations of Logic

- *Mathematical Logic* is a tool for working with elaborate *compound* statements.

- It includes:
- A formal 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.

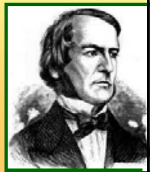
30-Jan-17

Foundations of Logic: Overview

- Propositional logic (§1.1-1.2):
  - Basic definitions. (§1.1)
  - Equivalence rules & derivations. (§1.2)
- Predicate logic (§1.3)
  - Predicates.
  - Quantified predicate expressions.
  - Equivalences & derivations.

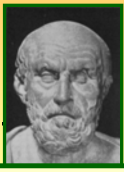
Propositional Logic (§1.1)

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



George Boole
(1815-1864)

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



Chrysippus of Soli
(ca. 281 B.C. – 205 B.C.)

Definition of a *Proposition*

- **Definition:** A *proposition* (denoted *p*, *q*, *r*, …) is simply:
- a *statement* (*i.e.*, a declarative sentence)
  - *with some definite meaning*, (not vague or ambiguous)
- having a *truth value* that's **either *true* (T) or *false* (F)**
  - it is **never** both, neither, or somewhere "in between!"
    - However, you might not *know* the actual truth value,
    - and, the truth value might *depend* on the situation or context.
- Later, we will study *probability theory,* in which we assign *degrees of certainty* ("between" **T** and **F**) 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"

- <u>But, the following are **NOT** propositions:</u>
- "Who's there?" (interrogative, question)
- "La la la la la." (meaningless interjection)
- "Just do it!" (imperative, command)
- "Yeah, I sorta dunno, whatever..." (vague)
- "1 + 2" (expression with a non-true/false value)

Few more

Propositions	Not Propositions
3 + 2 = 32	Bring me coffee!
CMSC 302 is Bryan's favorite class.	Is CMSC 302 Amy's favorite class?
Every cow has 4 legs.	3 + 2
There is other life in the universe.	Do you like Cake?

You now!
What sentences are not propositions?

(i) Paris is in France.
(ii) $1 + 1 = 2$.
(iii) $2 + 2 = 3$.
(iv) London is in Denmark.
(v) $9 < 6$.
(vi) $x = 2$ is a solution of $x^2 = 4$.
(vii) Where are you going?
(viii) Do your homework.

Well, you got it right

vii and viii ARE NOT!

3 are True, and 3 are False!

Which ones?

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 their 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 brown hair."
- then $\neg p$ = "I do **not** have brown hair."
- The *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* "∧" (*AND*) combines two propositions to form their logical *conjunction*.
- *E.g.* If
p="I will have salad for lunch." and
q="I will have steak for dinner.", then
$p \wedge q$="I will have salad for lunch **and** I will have steak for dinner."

∧ND

Remember: "∧" points up like an "A", and it means "AND"

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**.

AND can be algebraized as PROD or MIN

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

Operand columns: p, q

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

Which of the statements are TRUE?

(i) Paris is in France and $2 + 2 = 4$.
(ii) Paris is in France and $2 + 2 = 5$.
(iii) Paris is in England and $2 + 2 = 4$.
(iv) Paris is in England and $2 + 2 = 5$.

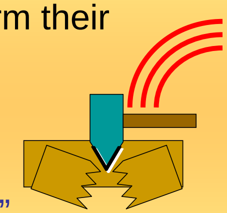
First only!

Why?

Well, see the table on a previous page

The Disjunction Operator

- The binary *disjunction operator* "$\vee$" (*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 "$\vee$" splits the wood, you can take 1 piece OR the other, or both.

Disjunction Truth Table

- Note that $p \vee q$ 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.
- Remark."¬" and "$\vee$" together are also universal.

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

Note difference from AND

OR can be algebraized as

SUM or MAX

Which of the statements are FALSE?

(i) Paris is in France or 2 + 2 = 4.
(ii) Paris is in France or 2 + 2 = 5.
(iii) Paris is in England or 2 + 2 = 4.
(iv) Paris is in England or 2 + 2 = 5.

Last only!

Why?

Nested Propositional Expressions

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

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* "⊕" (*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 in 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.
- Remark. "¬" 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.

In math terms, we can say XOR is true only if the sum = 1. E.g., take the second and fourth row: 0+1 = 1, means T, while 1+1 = 2, means F

Remark: The English word "or" is commonly used in two distinct ways. Sometimes it is used in the sense of "*p* or *q* or both", i.e., at least one of the two alternatives occurs, as above, and sometimes it is used in the sense of "*p* or *q* but not both", i.e., exactly one of the two alternatives occurs. For example, the sentence "He will go to Harvard or to Yale" uses "or" in the latter sense, called the *exclusive disjunction.* Unless otherwise stated, "or" shall be used in the former sense. This discussion points out the precision we gain from our symbolic language: $p \vee q$ is defined by its truth table and *always* means "*p* and/or *q*".

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.

XOR AND IMPLICATION OR

IDENTITY or BICONITIONAL

Table 3.1
Logic Functions of Two Variables

x_1	x_2	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}	f_{16}
0	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	0	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Table 3.1 is from V. Kecman's The MIT Press Book, 2001

The *Implication* Operator

antecedent consequent

- The *implication* $p \rightarrow q$ 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, meaning If you don't study hard, then either you will get a good grade, or a bad one, or you will fail)

Implication Truth Table

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

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

The only False case!

There is a simpler way to learn the IMPLICATION table than to memorize it.
It's based on THE EQUALITY

$p \to q$ **EQUALS** $\neg p \lor q$

30-Jan-17 **29/97**

Well, but can the IMPLICATION be expressed by NOT and AND?

- Sure it can and we start from the following equivalency:
- $\neg(p \to q)$ **EQUALS** $p \land \neg q$
- A one more negation of both sides leads to
- $p \to q$ **EQUALS** $\neg(p \land \neg q)$

This page is the answer to the question of one of your colleagues in the last class. It is not in your Fall 2016 slides, but it will be in the future ones ☺

30-Jan-17 **30/97**

Examples of Implications

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

30-Jan-17 **31/97**

Why does this seem wrong?

- Consider a sentence like,
  - "If I were woman, then dictators are democrats!"
- In logic, we consider the sentence **True** because we have **F** and **F** which gives **T**
- But, in normal English conversation, if I were to make this claim, you would think that I was lying.
  - Why this discrepancy between logic & language?

30-Jan-17 **32/97**

Cause of the Discrepancy

- In English, a sentence "if *p* then *q*" usually really *implicitly* means something like,
  - "<u>In all possible situations</u>, 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: $\forall s, P(s) \rightarrow Q(s)$
- In our example *the* previous slide's sentence is logically ***False***, because for me to be a women and for a Dictator to be a democrat is not a *possible* situation.

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*"
- We will see some equivalent logic expressions later.

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

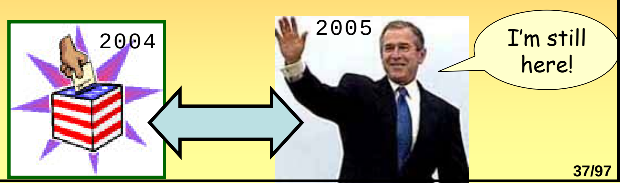
How do we know for sure?

- 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.
- Remark. This truth table is the exact **opposite** of ⊕'s!
- Thus, $p \leftrightarrow q$ means $\neg(p \oplus q)$
- $p \leftrightarrow q$ does **not** imply that p and q are true, or that either of them causes the other, or that they have a common cause.

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:	$\bar{p}$	pq	$+$	$\oplus$		
C/C++/Java (wordwise):	!	&&	||	!=		==
C/C++/Java (bitwise):	~	&	|	^		
Logic gates:						

Bits and Bit Operations

- 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",
- Also, ∧ = min, ∨ = MAX
  - You can find more in chapter 11.

Bit Strings

- A *Bit string* of *length* *n* is an ordered sequence (series, tuple) of $n \geq 0$ bits.
  - More on sequences in §3.2.
- By convention, bit strings are (sometimes) written left to right:
  - *e.g.* the "first" bit of the bit string "1001101010" is 1.
  - Watch out! Another common convention is that the rightmost bit is bit #0, the 2^{nd}-rightmost is bit #1, etc.
- When a bit string represents a base-2 number, by convention, the first (leftmost) bit is the *most significant* bit. *Ex.* $1101_2=8+4+1=13$.

Bitwise Operations

- Boolean operations can be extended to operate on bit strings as well as single bits.
- E.g.:
  01 1011 0110
  11 0001 1101

  11 1011 1111 Bit-wise OR, i.e. MAX

  01 0001 0100 Bit-wise AND, i.e. min

  10 1010 1011 Bit-wise XOR, i.e. MAX except for 1 1

End of §1.1

- You have learned about:
- Propositions: What they are.
- Propositional logic operators'
  - Symbolic notations.
  - English equivalents.
  - Logical meaning.
  - Truth tables.
- Atomic vs. compound propositions.
- Alternative notations.
- Bits and bit-strings.
- Next section: §1.2
  - Propositional equivalences.
  - How to prove them.

Propositional Equivalence

Propositional Equivalence (§1.2)

- 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*. (Here, we'll use truth table)

Tautologies and Contradictions

- A *tautology* is a compound proposition that is **true** *no matter what* the truth values of its atomic propositions are!
- *Ex.* $p \lor \neg p$ [What is its truth table?]
- A *contradiction* is a compound proposition that is **false** no matter what the truth values of its atomic propositions are!!
- *Ex.* $p \land \neg p$ [Truth table?]
  - Other compound propositions are ***contingencies***.

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.

Note! These 2 symbols are the 2 DIFFERENT symbols

In other words:

- 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

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

If ∨ =+, and ∧=*

- *Distributive*: $p \vee (q \wedge r) \Leftrightarrow (p \vee q) \wedge (p \vee r)$ Not working in math
  $p \wedge (q \vee r) \Leftrightarrow (p \wedge q) \vee (p \wedge r)$ Working in math
- *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: $p \leftrightarrow q \Leftrightarrow (p \rightarrow q) \wedge (q \rightarrow p)$
  $p \leftrightarrow q \Leftrightarrow \neg(p \oplus q)$

An Example Problem

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

$p \rightarrow q$ EQUALS $\neg p \vee q$

- $(p \wedge \neg q) \rightarrow (p \oplus r)$
- $\Leftrightarrow \neg(p \wedge \neg q) \vee (p \oplus r)$ [Expand definition of →]
- $\Leftrightarrow \neg(p \wedge \neg q) \vee ((p \vee r) \wedge \neg(p \wedge r))$ [Expand defn. of ⊕]
- $\Leftrightarrow (\neg p \vee q) \vee ((p \vee r) \wedge \neg(p \wedge r))$ [DeMorgan's Law]
- *cont.*

$p \oplus q \Leftrightarrow (p \vee q) \wedge \neg(p \wedge q)$
From the top of previous slide

Example Continued...

- $\Leftrightarrow (\neg p \vee q) \vee ((p \vee r) \wedge \neg(p \wedge r))$
- $\Leftrightarrow (q \vee \neg p) \vee ((p \vee r) \wedge \neg(p \wedge r))$ [∨ commutes]
- $\Leftrightarrow q \vee (\neg p \vee ((p \vee r) \wedge \neg(p \wedge r)))$ [∨ associative]
- $\Leftrightarrow q \vee (((\neg p \vee (p \vee r)) \wedge (\neg p \vee \neg(p \wedge r)))$ [distrib. ∨ over ∧

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

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

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

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

⇔ *cont.*

End of Long Example

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

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

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

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

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

⇔*Q.E.D.*

⇔*Remark.* *Q.E.D. (quod erat demonstrandum)*

(Which was to be shown.)

Review: Propositional Logic (§§1.1-1.2)

- 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. $p \Leftrightarrow q \Leftrightarrow r$ …

30-Jan-17 57/97

Predicate Logic

30-Jan-17 58/97

Predicate Logic (§1.3)

- *Predicate logic* is an extension of propositional logic that permits concisely reasoning about whole *classes* of entities.
- Propositional logic (recall) treats simple *propositions* (sentences) as atomic entities.
- In contrast, *predicate* logic distinguishes the *subject* of a sentence from its *predicate*.
  - Remember these English grammar terms?

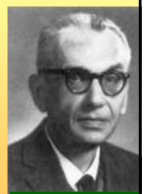
30-Jan-17 59/97

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!

30-Jan-17 60/97

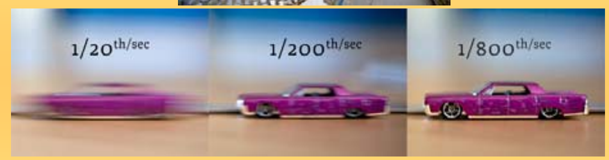
Other Applications



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 always remain *some* true statements that will *never be proven* by that procedure.
- *i.e.*, we can't discover *all* mathematical truths, unless we sometimes resort to making *guesses.*

Digression



Hence, it is not possible, buy using one sensor (technique, tool, algorithm, approach, camera shot) only, to get the information of both – Position and Velocity

We need at least two fast shutter shots + two time measurement + math algorithm which goes as

$$v = \frac{p_2 - p_1}{t_2 - t_1}$$

to calculate velocity

Practical Applications of Predicate Logic

- It is the basis for clearly expressed formal specifications for any complex system.
- It is basis for *automatic theorem provers* and many other Artificial Intelligence systems.
  - *E.g.* automatic program verification systems.
- Predicate-logic like statements are supported by some of the more sophisticated *database query engines* and *container class libraries*
  - these are types of programming tools.

In some texts, Predicate Logic is aka **Propositional Functions**

Let A be a given set. A *propositional function* (or: an *open sentence* or *condition*) defined on A is an expression

$$p(x)$$

which has the property that $p(a)$ is true or false for each $a \in A$. That is, $p(x)$ becomes a statement (with a truth value) whenever any element $a \in A$ is substituted for the variable x. The set A is called the *domain* of $p(x)$, and the set T_p of all elements of A for which $p(a)$ is true is called the *truth set* of $p(x)$. In other words,

$$T_p = \{x: x \in A,\ p(x) \text{ is true}\} \quad \text{or} \quad T_p = \{x: p(x)\}$$

Frequently, when A is some set of numbers, the condition $p(x)$ has the form of an equation or inequality involving the variable x.

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.**

Connect this with an English sentence: **The dog** **is** **sleeping**

S **W** **O**

Here, W & O are jointly called **PREDICATE**

In predicate logic, a *predicate* is modeled as a *function* $P(\cdot)$ from objects to propositions. Here, ***P*** means ***is sleeping***

$P(x)$ = "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).

- Remark. 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 $P(:)$ = " 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*.

Quantifier Expressions

- *Quantifiers* provide a notation that allows us to *quantify* (count) *how many* objects in the univ. of disc. satisfy a given predicate. Quantifier **defines i.e., binds** objects
- "∀" is the FOR∀LL or *universal* quantifier. ∀*x P*(*x*) means *for all* x in the u.d., *P* holds.

The symbol ∀ can be used to define the intersection

- "∃" is the ∃XISTS or *existential* quantifier. ∃*x P*(*x*) means there *exists* an *x* in the u.d. (that is, 1 or more) such that *P*(*x*) is true.

The symbol ∃ can be used to define the union

The Universal Quantifier ∀

- Example:
  Let the u.d. of x be parking spaces at the VC university.
  Let *P*(*x*) be the *predicate* "*x* is full."
  Then the *universal quantification of P*(*x*), ∀*x P*(*x*), is the *proposition:*
  - "All parking spaces at VCU are full."
  - *i.e.*, "Every parking space at VCU is full."
  - *i.e.*, "For each parking space at VCU, that space is full."

The Existential Quantifier ∃

- Example:
  Let the u.d. of x be parking spaces at the VCU.
  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 VCU is full."
  - "There is a parking space at VCU that is full."
  - "At least one parking space at VCU is full."

Free and Bound Variables

- An expression like *P*(*x*) is said to have a *free variable x* (meaning, *x* is undefined).
- A quantifier (either ∀ or ∃) *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*.

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: *e.g.* let $Q(y) = \forall x\, P(x,y)$

Which of these two bounded expressions is true

The proposition $(\forall n \in \mathbf{N})(n + 4 > 3)$ is **TRUE**

The proposition $(\forall n \in \mathbf{N})(n + 2 > 8)$ is **FALSE**

Similarly

The proposition $(\exists n \in \mathbf{N})(n + 4 < 7)$ is true since $\{n\colon n + 4 < 7\} = \{1, 2\} \neq \varnothing$.

The proposition $(\exists n \in \mathbf{N})(n + 6 < 4)$ is false since $\{n\colon n + 6 < 4\} = \varnothing$.

Negations on Quantifiers

- We will often want to consider the negation of a quantified expression.
- Example:
  - Consider the statement "Every student in the class has taken a course in calculus."
  - This statement is a **universal quantification**, namely, $\forall x\, P(x)$ where $P(x)$ is the statement "x has taken a course in calculus"

Negate a Universal Quantification

- The negation of the above statement is "It is not the case that every student has taken a course in calculus", namely, $\neg\forall x\, P(x)$.
- Or, put it another way, "There is at least a student in the class who has **not taken** a course in calculus", namely, $\exists x \neg P(x)$.
- This example illustrates the following equivalence:

$$\neg\forall x\, P(x) \equiv \exists x \neg P(x)$$

Negate an Existential Quantification

- Example:
  - Consider the statement "There is a student in the class who has taken a course in calculus", namely ∃x Q(x), where Q(x) is the statement "x has a course in calculus."
- The negation of this statement is the proposition "It is not the case that there is a student in the class who has taken a course in calculus", namely, ¬∃x Q(x).
- This is equivalent to "Every student in this class has not taken a course in calculus", namely, ∀x ¬Q(x).
- So

$$\neg\exists x\, Q(x) \equiv \forall x\, \neg Q(x)$$

Negation Equivalence

- De Morgan's Laws in the case of negations of quantifiers (assuming that all the elements of u.d. can be listed)

$\neg\forall x\, P(x) \equiv \neg(P(x_1) \wedge P(x_2) \wedge \cdots \wedge P(x_n))$

$\equiv \neg P(x_1) \vee \neg P(x_2) \vee \cdots \vee \neg P(x_n)$

$\equiv \exists x\, \neg P(x)$

$\neg\exists x\, P(x) \equiv \neg(P(x_1) \vee P(x_2) \vee \cdots \vee P(x_n))$

$\equiv \neg P(x_1) \wedge \neg P(x_2) \wedge \cdots \wedge \neg P(x_n)$

$\equiv \forall x\, \neg P(x)$

Examples on Negations

- What is the negations of the following statement?
  - "There is an honest politician"
- Solution:
  - "There is an honest politician" is represented by ∃x H(x) where H(x) is the statement "x is an honest politician"
  - The negation is "There is not a single honest politician" which is represented by ¬∃x H(x), or ∀x ¬H(x).

Examples on Negations (cont.)

- What are the negations of the following statements?
  - "All Canadians play hockey"
- Solution:
  - "All Canadians play hockey" is represented by ∀x H(x) where H(x) is the statement "x plays hockey"
  - The negation is "Some Canadian does not play hockey", which is represented by ∃x ¬H(x), or ¬∀x H(x).

Nesting of Quantifiers (§1.4)

- Example: Let the u.d. of x & y be people.
- Let $L(x,y)$="x likes y" (a predicate with 2 free variables)
- 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.)

30-Jan-17

Review: Predicate Logic (§1.3)

- 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.

30-Jan-17

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 (someone) soul whom *everyone* relies upon** (including himself)!
- $\exists x(\forall y\, R(x,y))$= **There's some needy person (someone) 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)!**

30-Jan-17

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.

30-Jan-17

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 $P(x)$." $=\exists x\ (x>0 \land P(x))$

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.

P(x):=x>1, Q(x):=x<10

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

Quantifier Equivalence Laws

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

DeMorgan's

Negations

- We can prove the laws:
  $\neg\forall x\ P(x) \Leftrightarrow \exists x\ \neg P(x)$
  $\neg\exists x\ P(x) \Leftrightarrow \forall x\ \neg P(x)$

More Equivalence Laws

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

30-Jan-17

Review: Predicate Logic (§1.3 & 1.4)

- 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*)."

30-Jan-17

More Notational Conventions

- Quantifiers bind as loosely as needed:
  parenthesize $\forall x\,(P(x) \wedge Q(x))$
- Consecutive quantifiers of the same type can be combined:
  $\forall x\, \forall y\, \forall z\, P(x,y,z) \Leftrightarrow$
  $\forall x,y,z\, P(x,y,z)$ or even
  $\forall xyz\, P(x,y,z)$
- All quantified expressions can be reduced to the canonical *alternating* form
  $\forall x_1 \exists x_2 \forall x_3 \exists x_4 \ldots P(x_1, x_2, x_3, x_4, \ldots)$

30-Jan-17

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*."

30-Jan-17

Examples

- Can predicate logic say "there exist at least two objects with property P"?
-
  Yes, that's easy:
  $\exists x\ \exists y\ (P(x) \wedge P(y) \wedge\ x \neq y)$

30-Jan-17

93/97

Examples ...

- Can predicate logic say "there exist exactly two objects with property P"?
-
  Yes:
  $\exists x\ \exists y\ (P(x) \wedge P(y) \wedge x \neq y\ \wedge$
  $\forall z\ (P(z) \rightarrow (z = x \vee z = y\ ))$

30-Jan-17

94/97

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.*

30-Jan-17

95/97

Deduction Example Continued

- <u>Some valid conclusions you can draw:</u>

- $H(\mathrm{s}) \rightarrow M(\mathrm{s})$ **[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.*

30-Jan-17

96/97

Another Example

- Definitions:
  $H(x)$:≡ "x is human";
  $M(x)$:≡ "x is mortal";
  $G(x)$:≡ "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.")

The Derivation

- $\forall x\, H(x) \rightarrow M(x)$ ("Humans are mortal") and
- $\forall x\, G(x) \rightarrow \neg M(x)$ ("Gods are immortal").
- $\neg\exists x\, (H(x) \wedge G(x))$ ("No human is a god.")

- $\forall x\, H(x)\rightarrow M(x)$ and $\forall x\, G(x)\rightarrow\neg M(x)$.
- $\forall x\, \neg M(x)\rightarrow\neg H(x)$ [Contrapositive.]
- $\forall x\, [G(x)\rightarrow\neg M(x)] \wedge [\neg M(x)\rightarrow\neg H(x)]$
- $\forall x\, G(x)\rightarrow\neg H(x)$ [Transitivity of →.]
- $\forall x\, \neg G(x) \vee \neg H(x)$ [Definition of →.]
- $\forall x\, \neg(G(x) \wedge H(x))$ [DeMorgan's law.]
- $\neg\exists x\, G(x) \wedge H(x)$ [An equivalence law.]
- $\neg\exists x\, (H(x) \wedge G(x))$ [commutativity]

End of §1.3-1.4, Predicate Logic

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

References

- Rosen
  Discrete Mathematics and its Applications, 6e
  Mc GrawHill, 2008