

Introduction to FoLT (Lecture-1)

- " FoLT aka Discrete Mathematics " is not the name of a branch of mathematics, like number theory, algebra, calculus, etc. Rather, it's a description of a set of branches of math that all have in common the feature that they are "discrete" rather than "continuous".
- The members of this set include (certain aspects of):
 - Logic and Boolean algebra
 - Set theory
 - Relations and Functions
 - Sequences and Series (or "sums")
 - Algorithms and Theory of Computation(Sem-IV)
 - Number Theory
 - Matrix Theory
 - Induction and Recursion
 - Counting and Discrete Probability
 - Graph Theory (including trees) (Sem-IV)

Application of FoLT

- Concepts and notations from **FoLT** are useful in studying and describing objects and problems in branches of computer science, such as
 - computer algorithms
 - programming languages
 - cryptography
 - automated theorem proving
 - software development.
 - AI

Unit-1: Foundations of Logic: Overview

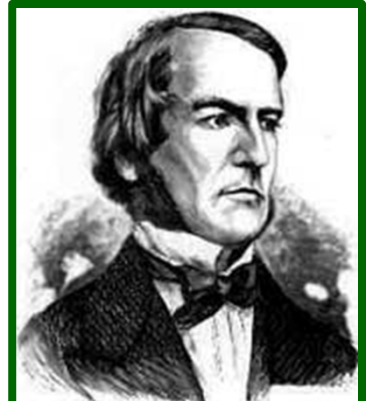
- Propositional logic:
 - Basic definitions.
 - Equivalence rules & derivations.
- Predicate logic
 - Predicates.
 - Quantified predicate expressions.
 - Equivalences & derivations.

Propositional Logic

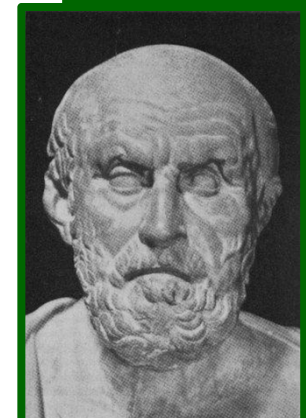
Propositional Logic is the logic of compound statements built from simpler statements using so-called *Boolean connectives*.

Some applications in computer science:

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



George Boole
(1815-1864)



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

Definition of a *Proposition*

Definition: A *proposition* (denoted p, q, r, \dots) is simply:

- a *statement* (*i.e.*, a declarative sentence[Not: imperative or interrogative])
 - *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.

Examples of Propositions

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

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

<u>Formal Name</u>	<u>Nickname</u>	<u>Arity</u>	<u>Symbol</u>
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* “ \neg ” (*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:

T \equiv True; F \equiv False
“ \equiv ” means “is defined as”

p	$\neg p$
T	F
F	T
Operand column	Result column

The Conjunction Operator

The binary *conjunction operator* “ \wedge ” (*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.”

Remember: “ \wedge ” points up like an “A”, and it means “AND”

Conjunction Truth Table

- Note that a conjunction $p_1 \wedge p_2 \wedge \dots \wedge p_n$ 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: \neg and \wedge operations together are sufficient to express *any* Boolean truth table!

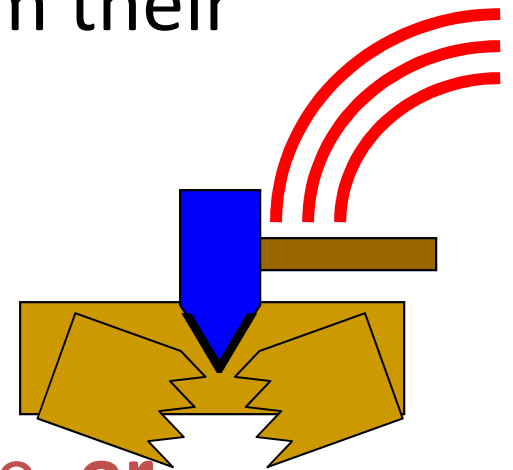
The Disjunction Operator

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.

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.
- “ \neg ” 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

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, “ \neg ” 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:

$\neg p$ =

$r \wedge \neg p$ = “It didn’t rain last night.”

$\neg r \vee p \vee q$ = “The lawn was wet this morning, and it didn’t rain last night.”

“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 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.
- “ \neg ” and “ \oplus ” 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.

The *Implication* Operator

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)

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!

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 Bush is president.”
True or False?
- “If the moon is made of green cheese, then I am richer than Bill Gates.” *True or False?*

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

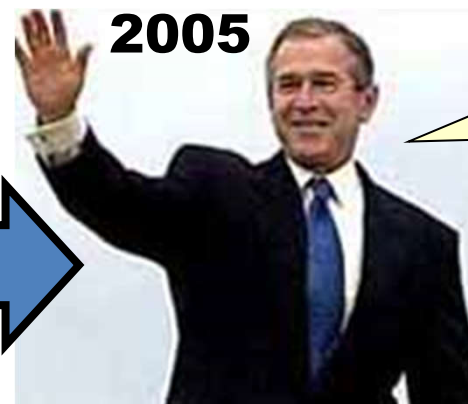
The *biconditional* operator

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.”



I'm still here!

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 that either of them causes the other, or that they have a common cause.

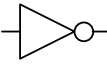

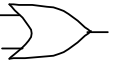
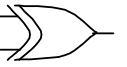
p	q	$p \leftrightarrow q$
F	F	T
F	T	F
T	F	F
T	T	T

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):	<code>!</code>	<code>& &</code>	<code> </code>	<code>!=</code>		<code>==</code>
C/C++/Java (bitwise):	<code>~</code>	<code>&</code>	<code> </code>	<code>^</code>		
Logic gates:						

Eg:1

What is the value of the variable x after the statement if $2 + 2 = 4$ then $x := x + 1$ if $x = 0$ before this statement is encountered? (The symbol $:=$ stands for assignment. The statement $x := x + 1$ means the assignment of the value of $x + 1$ to x .)

Eg-2

Solution: Because $2 + 2 = 4$ is true, the assignment statement $x := x + 1$ is executed. Hence, x has the value $0 + 1 = 1$ after this statement is encountered.

E.g-3

What are the contrapositive, the converse, and the inverse of the conditional statement “The home team wins whenever it is raining?”

Solution: Because “q whenever p” is one of the ways to express the conditional statement $p \rightarrow q$, the original statement can be rewritten as “If it is raining, then the home team wins.”

Consequently, the contrapositive of this conditional statement is “If the home team does not win, then it is not raining.”

The converse is “If the home team wins, then it is raining.”

The inverse is “If it is not raining, then the home team does not win.”

Only the contrapositive is equivalent to the original statement.

Eg:4

Construct the truth table of the compound proposition

$$(p \vee \neg q) \rightarrow (p \wedge q).$$

TABLE 7 The Truth Table of $(p \vee \neg q) \rightarrow (p \wedge q)$.

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

Precedence of Logical Operators

TABLE 8
**Precedence of
Logical Operators.**

<i>Operator</i>	<i>Precedence</i>
\neg	1
\wedge	2
\vee	3
\rightarrow	4
\leftrightarrow	5

e.g:5

How can this English sentence be translated into a logical expression?

“You can access the Internet from campus only if you are a computer science major or you are not a freshman.”

let a , c , and f represent “You can access the Internet from campus,”

“You are a computer science major,” and “You are a freshman,” respectively. Noting that “only if” is one way a conditional statement can be expressed, this sentence can be represented as

$$a \rightarrow (c \vee \neg f).$$