Izmir Institute of Technology

CENG 115

Discrete Structures

Slides are based on the Text
Discrete Mathematics & Its Applications **(6th Edition)**
by Kenneth H. Rosen

Slides were prepared by Dr. Michael P. Frank
for COT 3100 course in University of Florida

Module #1:
Foundations of Logic

Rosen 6th ed., § § 1.1-1.4

~80 slides, ~4-6 lectures

Foundations of Logic

Mathematical Logic is used to specify the meaning of mathematical statements.

The rules of logic are used to distinguish between valid and invalid mathematical arguments.

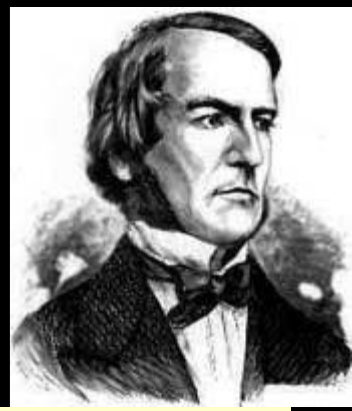
Since in this course we learn how to construct correct mathematical arguments, let us start with an introduction to logic.

Propositional Logic (§ 1.1)

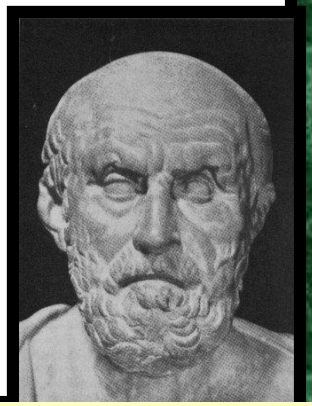
Propositional Logic is the logic of compound statements built from simpler statements using so-called *Boolean operators (AND, OR, NOT, etc.).*

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*

A *proposition* (*p*, *q*, *r*, …) is simply a *statement* (*i.e.*, a declarative sentence) with a *definite meaning*, having a *truth value* that's either *true* (T) or *false* (F) (**never** both, neither somewhere in between).

Examples:

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

What is not a Proposition?

The following are **NOT** propositions:

- "What time is it?" (question)
- "La la la la la." (meaningless interjection)
- "Just do it!" (imperative, command)
- "1 + 2" (expression with a non-true/false value)
- "$x + 2 = 3$" (since x is a variable, until assigning a value for x, expression has not a true/false value)

Operators

An *operator* combines one or more *operand* expressions into a larger expression which is called compound expression.

E.g.: 2+4 is a *compound expression* with one *operator* (+) and two *operands* (2 and 4).

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

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

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*.

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."

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

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."

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

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

Note difference from AND

A Simple Exercise

Let p="It rained last night",
q="The sprinklers were turned on last night"
r="The grass was wet this morning."

Translate each of the following into English:

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

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

$\neg r \vee p \vee q$ = "The grass wasn't wet this morning or it rained last night or the sprinklers were turned on last night."

The *Exclusive Or* Operator

The binary *exclusive-or operator* "⊕" (*XOR*) combines two propositions to form their logical "exclusive or".

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.

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

Note the 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." - $\oplus$

In English, we use the meaning of the sentence to distinguish between $\vee$ and $\oplus$

In this class, we assume "or" means inclusive.

The *Implication* Operator

antecedent consequence

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)

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 is sufficient for q"
- "q follows from p"
- "q is implied by p"

We may see some equivalent logic expressions later.

Implication Truth Table

- $p \rightarrow q$ is **false** only when p is true but q is **not** true.
- $p \rightarrow q$ does **not** guarantee that q is true.
  p may be T but q may be F.
- $p \rightarrow q$ does **not** require that p or q **are ever true**!
- *E.g.* "(1=0) → 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 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 Trump is the president." *True* or *False*?
- "If the moon is made of cheese, then I am richer than Bill Gates." *True* or *False*?

Why does this seem wrong?

- Consider a sentence like,
  - "If I am elected, I will lower taxes."
- In **logic**, we consider the sentence **True** as long as I am not elected, or I lower taxes.
- But in normal English conversation, if I were to make this claim and I am not elected, you would think the sentence is not true.
- Why this discrepancy between logic & language?

Resolving the Discrepancy

- For "If I am elected, I will lower taxes.", think in this way:
  - If the politician is elected and he/she lowers taxes then the statement is obviously **True**.
  - If the politician is not elected, the voters will not have any expectation that the politician will lower taxes. Therefore no matter what happens with the taxes, the statement is **True**.
  - It is only when the politician is elected but he/she does not lower taxes, the statement is **False** (True→False is False). Since the politician is elected, voters would expect this politician to lower taxes.

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?

Converse, Inverse, Contrapositive

$p \rightarrow q$: "If it is raining, then I will take subway".

- Its *converse* is: "If I take subway, then it is raining". $q \rightarrow p$.
- Its *inverse* is: "If it is not raining, then I will not take subway". $\neg p \rightarrow \neg q$.
- Its *contrapositive is*: "If I do not take subway, then it is not raining". $\neg q \rightarrow \neg p$.

One of these three has the *same meaning* with $p \rightarrow q$.

Which one? **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

Exercise

Compare the results of $p \rightarrow q$ and its inverse $\neg p \rightarrow \neg q$ using a truth table.

p	q	$\neg p$	$\neg q$	$p \rightarrow q$	$\neg p \rightarrow \neg q$
F	F	T	T	T	T
F	T	T	F	T	F
T	F	F	T	F	T
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 = "Trump wins the 2020 election."

q = "Trump will be president for all of 2021."

$p \leftrightarrow q$ = "If, and only if, Trump wins the 2020 election, Trump will be president for all of 2021."

Notice that both $(p \rightarrow q)$ and $(q \rightarrow p)$ are true.

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!

  $p \leftrightarrow q$ means $\neg(p \oplus q)$
- $p \leftrightarrow q$ does **not** imply that p and q are true.
- p IFF q is equivalent to $(p \rightarrow q) \wedge (q \rightarrow p)$.

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

Universal set of Boolean operators

- When a set of operators over a given domain is sufficient to express *any* possible operator in that domain, it is called a universal set.
- ¬ and ∧ operators together are (universal) sufficient to express *any* Boolean operator!
- ¬ and ∨ together are also universal.

E.g. $p \rightarrow q$ *is equal to* $\neg p \vee q$

$p \leftrightarrow q$ *is equal to* $(p \rightarrow q) \wedge (q \rightarrow p)$

and $(\neg p \vee q) \wedge (\neg q \vee p)$ *and* $\neg(p \wedge \neg q) \wedge \neg(q \wedge \neg p)$.

Boolean Operations Summary

We have seen 1 unary operator and 5 binary operators. 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

Precedence of Logical Operators

We generally use parentheses to specify the order, but if we don't, precedence rules apply:

Operator	*Precedence*
$\neg$	1
$\wedge$	2
$\vee$	3
$\rightarrow$	4
$\leftrightarrow$	5

Exercise

- Show by truth table that $\neg(p \vee q)$ and $\neg p \wedge \neg q$ are equivalent.

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

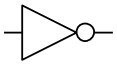
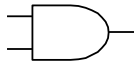
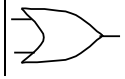
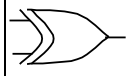
Exercise: Translating Sentences

- q is "You can ride roller coaster"
- r is "You are under 4 feet tall"
- s is "You are older than 16 years old"

How do you express "You cannot ride roller coaster if you are under 4 feet tall and you are under 16 years old" in logic?

Answer: $(r \wedge \neg s) \rightarrow \neg q$

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".

p	q	$p \wedge q$	$p \vee q$	$p \oplus q$
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

Bit Strings

- A *Bit string* of *length n* is an ordered series or sequence of $n \geq 0$ bits.
- When a bit string represents a base-2 number, by convention the first bit is the *most significant* bit. *Ex.* $1101_2 = 8+4+1 = 13$.

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
  Bit-wise AND
  Bit-wise XOR

End of § 1.1

You have learned about:

- Propositions: What they are.
- Propositional logic operators'
  - Symbolic notations.
  - English equivalents.
  - Logical meaning.
  - Truth tables.

- Compound propositions.
- Alternative notations.
- Bits and bit-strings.
- Next section: § 1.2
  - Propositional equivalences.
  - How to prove them.

Propositional Equivalence (§ 1.2)

Two *syntactically* (*i.e.*, textually) different compound propositions may be *semantically* identical (*i.e.*, have the same meaning).

We call them *logically equivalent*.

We will learn:

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

Tautologies and Contradictions

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

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

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

Other compound props. 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 true.

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 all rows of their truth tables.

Proving Equivalence via Truth Tables

Exercise: 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 laws are similar to the arithmetic laws you may have learned in algebra, but for propositional equivalences instead.
- They are used to match a complicated propositions and to find equivalence for them.

Equivalence Laws - Examples

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

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

Exercise: Check whether the following proposition is a tautology or not

$$\{(p \vee \neg q) \wedge (\neg p \wedge \neg q)\} \vee q$$

a) Using a truth table

b) Using the derivation rules

An Example Problem

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

Solution:

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

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

[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..*

Example Continued...

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

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

End of Long Example

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

[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 (§§ 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. $p \Leftrightarrow q \Leftrightarrow r$ …