Discrete Mathematics

Lecture 1

Introduction to Propositional Logic,

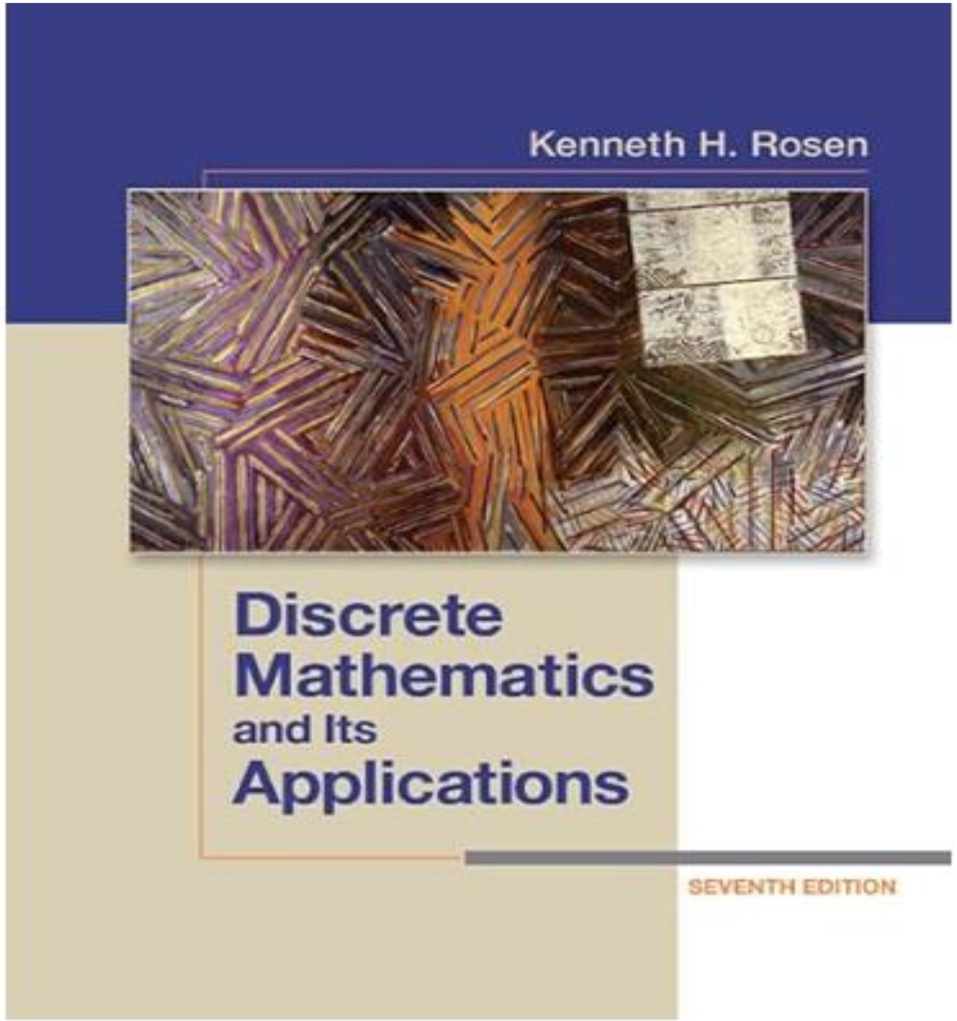
Compound Propositions

By

Asst. Lect. Ali Al-khawaja

Lectures Reference

- https://faculty.ksu.edu.sa/sites/default/files/rosen_discrete_mathematics_and_its_applications_7th_edition.pdf



Goals of a Discrete Mathematics Course

- Learn how to think mathematically.
- Grasp the basic logical and reasoning mechanisms of mathematical thought.
- Acquire logic and proof as the basics for abstract thinking.
- Improve problem-solving skills.
- Grasp the basic elements of induction, recursion, combination and discrete structures.

Discrete Mathematics is a Gateway Course

Topics in discrete mathematics will be important in many courses that you will take in the future:

- **Computer Science**: Computer Architecture, Data Structures, Algorithms, Programming Languages, Compilers, Computer Security, Databases, Artificial Intelligence, Networking, Graphics, Game Design, Theory of Computation, ……
- **Mathematics:** Logic, Set Theory, Probability, Number Theory, Abstract Algebra, Combinatorics, Graph Theory, Game Theory, Network Optimization, …
- **Other Disciplines:** You may find concepts learned here useful in courses in philosophy, economics, linguistics, and other departments.

Course Syllabus

- The Foundations: Logic and Proofs
- Basic Structures: Sets, Functions, Sequences, and Sums
- Algorithms
- Induction and Recursion
- Graphs
- Trees

The Foundations: Logic and Proofs

- Introduction to Propositional Logic.
- Compound Propositions.
- Applications of Propositional Logic.
- Propositional Equivalences.
- Predicates and Quantifiers.
- Arguments.
- Proofs Techniques.

Introduction to Propositional Logic

❖ What is Logic?

- Logic is the discipline that deals with the methods of reasoning.

- On an elementary level, logic provides rules and techniques for determining whether a given argument is valid.

- Logical reasoning is used in mathematics to prove theorems.

Introduction to Propositional Logic

- The basic building blocks of logic is **Proposition**
- A proposition (or statement) is a **declarative sentence** that is either **true** or **false**, but **not both**.
- The area of logic that deals with propositions is called **propositional logics**.

Examples:

Propositions	Truth value
$2 + 3 = 5$	True
$5 - 2 = 1$	False
Today is Friday	False
$x + 3 = 7$, for $x = 4$	True
Cairo is the capital of Egypt	True

Sentences	Is a Proposition
What time is it?	**Not** propositions
Read this carefully.	**Not** propositions
$x + 3 = 7$	**Not** propositions

Introduction to Propositional Logic

- We use letters to denote propositional variables $\boldsymbol{q}$, $\boldsymbol{r}$, $\boldsymbol{s}$, $\boldsymbol{t}$, …
- The truth value of a proposition is true, denoted by T, if it is a true proposition and false, denoted by F, if it is a false proposition.

Compound Propositions

- Compound Propositions are formed from existing propositions using logical operators.

Compound Propositions

❖ Negation

DEFINITION 1

Let p be a proposition. The *negation of p*, denoted by $\neg p$ (also denoted by $\overline{p}$), is the statement

"It is not the case that p."

The proposition $\neg p$ is read "not p." The truth value of the negation of p, $\neg p$, is the opposite of the truth value of p.

Compound Propositions

Example: Solution

Find the negation of the proposition

p: "Cairo is the capital of Egypt"

The negation is

$\neg p$: "It is not the case that Cairo is the capital of Egypt"

This negation can be more simply expressed as

$\neg p$: "Cairo is not the capital of Egypt"

Compound Propositions

Example: Solution

Find the negation of the proposition

p: "Cairo is the capital of Egypt"

The negation is

$\neg p$: "It is not the case that Cairo is the capital of Egypt"

This negation can be more simply expressed as

$\neg p$: "Cairo is not the capital of Egypt"

Compound Propositions

❖ Truth Table

- Truth Table: is a table that gives the truth values of a compound statement.

The Truth Table for the Negation of a Proposition

p	$\neg p$
T	F
F	T

Proposition

Truth Values

Logical Connectives

DEFINITION 2

Let p and q be propositions. The *conjunction* of p and q, denoted by $p \wedge q$, is the proposition "p and q." The conjunction $p \wedge q$ is true when both p and q are true and is false otherwise.

Example

p: Today is Friday.

q: It is raining today.

$p \wedge q$: Today is Friday and it is raining today.

TABLE 2 **The Truth Table for the Conjunction of Two Propositions.**

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

Logical Connectives

DEFINITION 3

Let p and q be propositions. The *disjunction* of p and q, denoted by $p \vee q$, is the proposition "p or q." The disjunction $p \vee q$ is false when both p and q are false and is true otherwise.

Example

p: Today is Friday.

q: It is raining today.

$p \vee q$: Today is Friday or it is raining today.

TABLE 3 The Truth Table for the Disjunction of Two Propositions.

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

Logical Connectives

DEFINITION 4

Let p and q be propositions. The *exclusive or* of p and q, denoted by $p \oplus q$ (or p XOR q), is the proposition that is true when exactly one of p and q is true and is false otherwise.

Example

p : They are parents.

q : They are children.

$p \oplus q$: They are parents or children but not both.

TABLE 4 **The Truth Table for the Exclusive Or of Two Propositions.**

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

Logical Connectives

DEFINITION 5

Let p and q be propositions. The *conditional statement* $p \rightarrow q$ is the proposition "if p, then q." The conditional statement $p \rightarrow q$ is false when p is true and q is false, and true otherwise. In the conditional statement $p \rightarrow q$, p is called the *hypothesis* (or *antecedent* or *premise*) and q is called the *conclusion* (or *consequence*).

"if p, then q"
"if p, q"
"p is sufficient for q"
"q if p"
"q when p"
"a necessary condition for p is q"
"q unless $\neg p$"

TABLE 5 The Truth Table for the Conditional Statement $p \rightarrow q$.

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

"p implies q"
"p only if q"
"a sufficient condition for q is p"
"q whenever p"
"q is necessary for p"
"q follows from p"

Logical Connectives

EXAMPLE 1

"If you get 100% on the final, then you will get an A."

If you manage to get a 100% on the final, then you would expect to receive an A. If you do not get 100% you may or may not receive an A depending on other factors. However, if you do get 100%, but the professor does not give you an A, you will feel cheated.

Logical Connectives

EXAMPLE 2

Let p be the statement "Maria learns discrete mathematics" and q the statement "Maria will find a good job." Express the statement $p \rightarrow q$ as a statement in English.

"If Maria learns discrete mathematics, then she will find a good job."

"Maria will find a good job when she learns discrete mathematics."

Logical Connectives

EXAMPLE 3

"If today is Friday, then $2 + 3 = 6$."

is true every day except Friday, even though $2 + 3 = 6$ is false.

Logical Connectives

DEFINITION 6

Let p and q be propositions. The *biconditional statement* $p \leftrightarrow q$ is the proposition "p if and only if q." The biconditional statement $p \leftrightarrow q$ is true when p and q have the same truth values, and is false otherwise. Biconditional statements are also called *bi-implications*.

TABLE 6 **The Truth Table for the Biconditional $p \leftrightarrow q$.**

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

"p is necessary and sufficient for q"
"if p then q, and conversely"
"p iff q." "p exactly when q."

"You can take the flight if and only if you buy a ticket."

Truth Tables of Compound Propositions

EXAMPLE 1

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.

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

Truth Tables of Compound Propositions

EXAMPLE 2

Construct the truth table of the compound proposition $(p \wedge \neg q) \rightarrow r$

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

Truth Tables of Compound Propositions

EXAMPLE 2

Construct the truth table of the compound proposition $(p \wedge \neg q) \rightarrow r$

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

Truth Tables of Compound Propositions

EXAMPLE 2

Construct the truth table of the compound proposition $(p \wedge \neg q) \rightarrow r$

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

Truth Tables of Compound Propositions

EXAMPLE 2

Construct the truth table of the compound proposition $(p \wedge \neg q) \to r$

p	q	r	$\neg q$	$p \wedge \neg q$	$(p \wedge \neg q) \to r$
T	T	T	F	F	
T	T	F	F	F	
T	F	T	T	T	
T	F	F	T	T	
F	T	T	F	F	
F	T	F	F	F	
F	F	T	T	F	
F	F	F	T	F	

Truth Tables of Compound Propositions

EXAMPLE 2

Construct the truth table of the compound proposition $(p \wedge \neg q) \rightarrow r$

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

Logic and Bit Operations

- Computers represent information using bits. A bit is a symbol with two possible values, namely, 0 (zero) and 1 (one).

Truth Value	*Bit*
T	1
F	0

Computer Bit Operations

- We will also use the notation OR, AND, and XOR for the operators ∨, ∧, and ⊕, as is done in various programming languages.

TABLE 9 **Table for the Bit Operators *OR*, *AND*, and *XOR*.**

x	y	$x \vee y$	$x \wedge y$	$x \oplus y$
0	0	0	0	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	0

Bit Strings

- Information is often represented using bit strings, which are lists of zeros and ones. When this is done, operations on the bit strings can be used to manipulate this information.

A *bit string* is a sequence of zero or more bits. The *length* of this string is the number of bits in the string.

101010011 is a bit string of length nine.

Example

- Find the bitwise OR, bitwise AND, and bitwise XOR of the bit strings 01 1011 0110 and 11 0001 1101

01 1011 0110	
11 0001 1101	
11 1011 1111	bitwise *OR*
01 0001 0100	bitwise *AND*
10 1010 1011	bitwise *XOR*