Welcome to the course

Discrete Mathematics and Its Applications

Instructor: Lin, Lanfen（林兰芬）

E-mail: llf@zju.edu.cn

Tel: 87952699

Lectures: Monday 1-2

Wednesday 3-4

TA: 郑寒

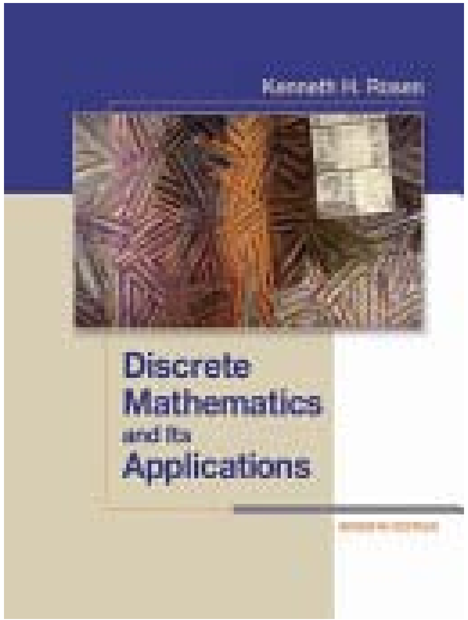
QQ群: 703576737

Ftp: ftp://dm:dm@10.214.145.163:22

Text Book

Discrete Mathematics and Its Applications
(Seventh Edition)

Author: Kenneth H.Rosen
Publisher: McGraw-Hill
机械工业出版社



🕮What is Discrete Mathematics?

- **The part of mathematics devoted to the study of discrete (as opposed to continuous) objects and their relation.**
  - Calculus deals with continuous objects and is not part of discrete mathematics.
  - Examples of discrete objects: integers, steps taken by a computer program, distinct paths to travel from point A to point B on a map along a road network, …
- **A course provides the mathematical background needed for**
  - all subsequent courses in computer science
  - all subsequent courses in the many branches of discrete mathematics.
- **A gateway to many courses that you will take in the future**

  For examples:
  - **Computer Science: Data Structures, Algorithms, Databases, Artificial Intelligence, Graphics,……**
  - **Mathematics: Logic, Set Theory, Probability, Number Theory, Abstract Algebra, Combinatorics, Graph Theory,…..**

Framework of the Course

- Logic & Reasoning
- Set Theory
- The Foundation of Algorithms
- Combinatorial Theory (Counting)
- Relations & Graph Theory
- Algebra System

Goals of the Course

- Logic & Reasoning
- Set Theory
- The Foundation of Algorithms
- Combinatorial Theory (Counting)
- Relations & Graph Theory
- Algebra System

Mathematical Reasoning

- **Develop the ability to read, understand, and construct mathematical arguments and proofs.**

Combinatorial Analysis

- **Learn how to solve counting problems**
- **It's an important problem-solving skill**

Goals of the Course

- Logic & Reasoning
- Set Theory
- The Foundation of Algorithms
- Combinatorial Theory (Counting)
- Relations & Graph Theory
- Algebra System

Discrete Structures

- To learn how to work with discrete structures

  Examples:

  - sets, permutations, relations, graphs, trees, and finite state machines.

Algorithmic Thinking

- Develop the ability of algorithmic thinking involves
  - specifying algorithms,
  - analyzing the required memory and time
  - verifying algorithm

Goals of the Course

- Logic & Reasoning
- Set Theory
- The Foundation of Algorithms
- Combinatorial Theory (Counting)
- Relations & Graph Theory
- Algebra System

Applications and Modeling

- **To appreciate and understand the wide range of applications of the topics in discrete mathematics and develop the ability to develop new models in various domains.**
- **Concepts from discrete mathematics have not only been used to address problems in computing, but have been applied to solve problems in many areas such as chemistry, biology, linguistics, geography, business, etc.**

Other

- **Homework assignments、Quiz、Final examination、Small report for exploration of graph theory and its application**

- **Grading**

Final examination	:	60%
Quiz (3)	:	27%
Report	:	6%
Assignments	:	7%

The Foundations: Logic and Proofs

Chapter 1, Part I: Propositional Logic

Chapter Summary

✓Propositional Logic

- The Language of Propositions
- Applications
- Logical Equivalences

✓ Predicate Logic

- The Language of Quantifiers
- Logical Equivalences
- Nested Quantifiers

✓ Proofs

- Rules of Inference
- Proof Methods
- Proof Strategy

Propositional Logic Summary

✓ **The Language of Propositions**

- **Connectives**
- **Truth Values**
- **Truth Tables**

✓ **Applications**

- **Translating English Sentences**
- **System Specifications**
- **Logic Puzzles**
- **Logic Circuits**

✓ **Logical Equivalences**

- **Important Equivalences**
- **Showing Equivalence**
- **Satisfiability**

Propositional Logic

Section 1.1

Section Summary

- ✓**Propositions**
- ✓**Connectives**
  - **Negation**
  - **Conjunction**
  - **Disjunction**
  - **Exclusive Or**
  - **Implication; contrapositive, inverse, converse**
  - **Biconditional**
- ✓**Truth Tables**

Propositions

A *proposition* is a declarative sentence that is either true or false, but not both.

- **The *truth value* of a proposition： T(1)， F(0)**

〖Example 〗

Consider the following sentences.

① The Olympic Games was held in Beijing in 2008.

② The integer 9 is prime.

③ This statement is false.

④ Please open the book.

⑤ What time is it?

⑥ $x + 1 = 4$.

Propositional Logic

- **Propositional logic (calculus): the area of logic that deals with proposition.**
- **Constructing Propositions**
  - **Propositional Variables: *p, q, r, s, …***
  - **The proposition that is always true is denoted by T and the proposition that is always false is denoted by F.**
  - **Compound Propositions: constructed from logical connectives and other propositions**
    - ✓ **Negation (NOT)**
    - ✓ **Conjunction (AND)**
    - ✓ **Disjunction (OR)**
    - ✓ **Exclusive or (XOR)**
    - ✓ **Implication (if – then)**
    - ✓ **Biconditional (if and only if)**

1. Negation (NOT)

The negation of a proposition *p* : ¬*p* (not *p*)

True when *p* is false, false when *p* is true.

p	¬p
T	F
F	T

The *truth table* for the negation of a proposition.

〖Example〗

If *p* denotes “The earth is round.”, then ¬*p* denotes “It is not the case that the earth is round,” or more simply “The earth is not round.”

2. Conjunction (AND)

The conjunction of propositions p and q : $p \wedge q$ (*p* and *q*)

True when both p and q are true.

p	q	p ∧ q
T	T	T
T	F	F
F	T	F
F	F	F

〖Example〗

1. If p denotes "I am at home." and q denotes "It is raining." then $p \wedge q$ denotes "I am at home **and** it is raining."
2. The sun is shining , **but** it is raining.

3. Disjunction (OR)

The disjunction of propositions *p* and *q* : $p \vee q$ **(** p **or** q**)**

False when both *p* and *q* are false.

p	q	p ∨ q
T	T	T
T	F	T
F	T	T
F	F	F

〖Example〗

If *p* denotes "I am at home." and *q* denotes "It is raining." then $p \vee q$ denotes "I am at home or it is raining."

4. Exclusive Or (XOR)

the exclusive or of *p* and *q*: $p \oplus q$:

True when exactly one of *p* and *q* is true.

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

〖Example 〗

If *p* denotes "Today is Tuesday." **and *q* denotes** "I will go to the beach." **then** $p \oplus q$ **denotes "Either** today is Tuesday **or** I will go to the beach."

Note：

In English "or" has two distinct meanings.

- **Inclusive or:** $\vee$
- **Exclusive or:** $\oplus$

〖Example〗 How can the following sentence be translated into a logical expression?

(1) Students who have taken calculus or computer science can take this class.

(2) George Boole was born in 1815 or 1816.

(3) "Soup or salad comes with this entrée,"

5. Implication (if - then)

Implication or conditional statement : $p \to q$ (if p then q)

False when p is true and q is false.

p	q	p →q
T	T	T
T	F	F
F	T	T
F	F	T

〖Example 〗

If p denotes "I am at home." and q denotes "It is raining." then $p \to q$ denotes "If I am at home then it is raining."

In $p \to q$, p is the hypothesis (antecedent or premise) and q is the conclusion (or consequence).

Understanding Implication

- **In $p \rightarrow q$ there does not need to be any connection between the antecedent or the consequent. The "meaning" of $p \rightarrow q$ depends only on the truth values of p and q.**
- **These implications are perfectly fine, but would not be used in ordinary English.**
  - ***If the moon is made of green cheese, then I have more money than Bill Gates.***
  - ***If today is Monday, then 1+1=2.***
  - ***If today is Monday, then 1+1=3.***

Understanding Implication

- **One way to view the logical conditional is to think of an obligation or contract.**

Example: If I am elected, then I will lower taxes.

- **If the politician is elected and does not lower taxes, then the voters can say that he or she has broken the campaign pledge.**
- **This corresponds to the case where p is true and q is false.**

Something similar holds for the professor.

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

Different Ways of Expressing $p \to q$

- **if** p**, then** q
- **if** p, q
- p **is sufficient for** q
- p **implies** q
- p **only if** q
- **a sufficient condition for** q **is** p

- q **if** p
- q **when** p
- q **follows from** p
- q **unless** $\neg p$
- q **whenever** p
- q **is necessary for** p
- **a necessary condition for** p **is** q

Converse, Contrapositive, and Inverse

- From $p \rightarrow q$ we can form new conditional statements .
  - $q \rightarrow p$ is the **converse** of $p \rightarrow q$
  - $\neg q \rightarrow \neg p$ is the **contrapositive** of $p \rightarrow q$
  - $\neg p \rightarrow \neg q$ is the **inverse** of $p \rightarrow q$

Example: Find the converse, inverse, and contrapositive of "It raining is a sufficient condition for my not going to town."

Solution:

converse: If I do not go to town, then it is raining.

inverse: If it is not raining, then I will go to town.

contrapositive: If I go to town, then it is not raining.

The **contrapositive has the same truth values as the original implication.**

6. Biconditional (if and only if)

the biconditional proposition $p \leftrightarrow q$ (p if and only if q)

True when p **and** q **have the same truth values.**

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

〖Example〗

If p **denotes "I am at home." and** q **denotes "It is raining."**
then $p \leftrightarrow q$ **denotes "I am at home if and only if it is raining."**

Expressing the Biconditional

- **Some alternative ways "p if and only if q" is expressed in English:**
  - ***p* is necessary and sufficient for *q***
  - **if *p* then *q* and conversely**
  - ***p* iff *q***

Truth Table for Compound Propositions

Construction of a truth table:

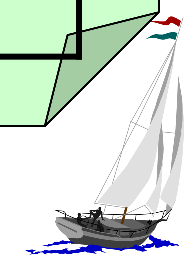
- **Rows**
  - Need a row for every possible combination of values for the atomic propositions.
- **Columns**
  - Need a column for the compound proposition (usually at far right)
  - Need a column for the truth value of each expression that occurs in the compound proposition as it is built up.
    - This includes the atomic propositions

Example Truth Table

Construct the truth value table for $(p \wedge q) \to r$

Solution:

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



Equivalent Propositions

- **Two propositions are equivalent if they always have the same truth value.**

Example: Show using a truth table that the implicaion is equivalent to the contrapositive.

Solution:

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

Using a Truth Table to Show Non-Equivalence

Example:

Show using truth tables that neither the converse nor inverse of an implication are not equivalent to the implication.

Solution:

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

Two Problemes

① **How many rows are there in a truth table with n propositional variables?**

Solution: 2^n We will see how to do this in Chapter 6.

② **Note that this means that with *n* propositional variables, we can construct (   ) distinct (i.e., not equivalent) propositions.**

Precedence of Logical Operators

- Parentheses gets the highest precedence
- Then $\neg \quad \wedge \quad \vee \quad \rightarrow \quad \leftrightarrow$

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

For example,

$p \wedge q \vee r$ **means** $(p \wedge q) \vee r$, **not** $p \wedge (q \vee r)$

$p \vee q \rightarrow r$ **means** $(p \vee q) \rightarrow r$

Logic and Bit Operations

A ***Boolean variable*** **is one whose value is either true or false.**

Bit**: has two possible values , namely, 0 and 1**

A Boolean variable can be represented using a bit.

Computer ***bit operations*** **correspond to logical operations of Boolean variables.**

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

A *bit string* is a sequence of zero or more bits.
The *length* of this string is the number of bits in the string.
***Bitwise operations* are bit operations extended to bit strings.**

The bitwise of two strings of the same length:

- **Bitwise OR**
- **Bitwise AND**
- **Bitwise XOR**

〖Example 〗 Find the bitwise OR, bitwise AND, and bitwise XOR of the bit strings 01 1011 0110 and 11 0001 1101.

Solution:

The bitwise OR, bitwise AND, and bitwise XOR of these strings are obtained by taking the OR, AND, and XOR of the corresponding bits, respectively.

This gives us

01 1011 0110

11 0001 1101

----

11 1011 1111 bitwise OR

01 0001 0100 bitwise AND

10 1010 1011 bitwise XOR

Homework:

Seventh Edition:

P.13 8(f, g, h), 14, 16, 37(c)