Discrete Mathematics and Statistics - CPT107

Xi'an Jiaotong-Liverpool University

西交利物浦大学

2023–2024

Module Organizers

- Prof. Ka Lok MAN (Module Leader)
- SD437
- Email: ka.man@xjtlu.edu.cn

- Dr. Gabriela Mogos (Co-Module Leader Teaching)
- Will start after Week 7/Week 8

Module Introduction

CPT107, Discrete Mathematics and Statistics

Aims of the Module

- To introduce the notation, terminology, and techniques underpinning the discipline of Theoretical Computer Science.
- To provide the mathematical foundation necessary for understanding datatypes as they arise in Computer Science and for understanding computation.
- To introduce the basic proof techniques which are used for reasoning about data and computation.
- To introduce the basic mathematical tools needed for specifying requirements and programs, and for analysing algorithms.

Organization of the Module

- Lectures
- Tutorials
- Continuous Assessments
- Textbooks
- Office hours

Lectures and Tutorials

We will have two lectures and/or tutorials per week (two hours each).

Special thanks go to Prof. *Leslie Ann Goldberg* for sharing the teaching materials.

CPT107 Students, please:

- Get copies of the slides from Learning Mall. Read the slides before (and after) the lecture.
- Take notes. (University is a lot different from high school.)
- Class problems and be active learner.

Tutorials

- Teaching assistants will also help to coordinate the tutorial for this module.

- Tutorial is once per week (in general).
  The first tutorial will be in Week 2.

- Try to solve the problems by yourself first. Solutions will be presented and discussed during tutorials.

Assessments & Examination

- Assessment 1 (10%). Week 7/8
  - . Written assignment.
- Assessment 2 (10%). Week 12/13/14.
  - Written assignment.
- Examination (80%).
  - Written examination.
- Summer resit examination (100%).

Textbooks/Reference books

- Rod Haggerty, Discrete Mathematics for Computing, Pearson, 2002/later edition.
- John Truss, Discrete Matematics for Computer Scientists, Addison Wesley, 1999/later edition.
- [Newer one] Susanna S. Epp, Discrete Mathematics with Applications, Brooks Cole, 4th edition, 2010/later edition.

amazon Try Prime

International Shipping | Discrete Mathematics for Computing

Deliver to

Departments | Your Amazon.com | Today's Deals | Gift Cards | Registry | Sell | Treasure Truck

1-16 of 428 results for **International Shipping : "Discrete Mathematics for Computing"**

Show results for

Books
- Discrete Mathematics
- Computer Science

Kindle Store
- Discrete Mathematics
- Computers & Technology

See All 3 Departments

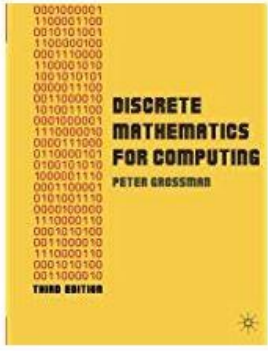
Oxford University Pre...

OXFORD UNIVERSITY PRESS

Short introductions to all your new courses

$9.42 prime

$11.90 prime



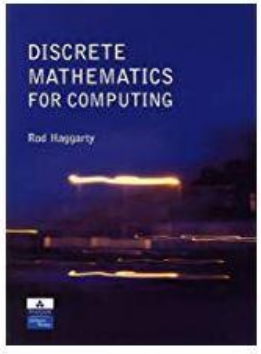
Discrete Mathematics for Computing Dec 16, 2008

by Peter Grossman

$36.44 ~~$69.99~~

Only 1 left in stock - order soon.

More Buying Choices
$29.15 (37 used & new offers)



Discrete Mathematics for Computing Jan 15, 2002

by Rod Haggarty

Eligible for Shipping to China

$52.27 ~~$67.00~~

Only 1 left in stock - order soon.

More Buying Choices
$22.00 (24 used & new offers)

See newer edition of this book ›

Course Contents

- Part 1. Number Systems and Proof Techniques
- Part 2. Set Theory
- Part 3. Relations
- Part 4. Functions
- Part 5. Propositional Logic and First-Order Predicate Logic
- Part 6. Combinatorics, Discrete probability and Statistics

Number Systems and Proof Techniques

- The most basic datatypes
  - Natural Numbers
  - Integers
  - Rationals
  - Real Numbers
  - Prime Numbers
- Proof Techniques
  - Finding a counter-example
  - Proof by contradiction
  - Proof by induction

Datatypes

A datatype in a programming language is a set of values and the operations on those values. The datatype states:

- the possible values for the datatype
- the operations that can be performed on the values
- the way that values are stored

Number Systems and Proof Techniques

- The most basic datatypes
  - Natural Numbers
  - Integers
  - Rationals
  - Real Numbers
  - Prime Numbers

Number Systems and Proof Techniques

- Proof Techniques
  - Finding a counter-example
  - Proof by contradiction
  - Proof by induction

These are used to reason about data types and to reason about algorithms.

An algorithm is a method for solving a problem using a sequence of instructions. (It is the idea that makes a computer program work.)

We use proof techniques, both to show that an algorithm is correct and to show that it is efficient.

Examples

- For $x, y, z \in \mathbb{Z}$, use proof by contradiction to show the following statement: if $x^2 + y^2 = z^2$, then at least one of x and y is even.

- For all positive integers n, use proof by induction to show that:

$$2 + 5 + 8 + \cdots + (3n - 1) = \frac{n}{2}(3n + 1).$$

Set Theory

The datatypes available in programming langauges are built from simple datatypes using sets.

- Notation for sets.
- What is a subset of a set?
- When are two sets equal?
- Operations on sets (how to construct new sets from old sets)
- Algebra of sets.
- Cardinality of sets.
- The cartesian product of sets.
- Bit strings.

Set Theory and Relations

Databases: Most databases store information as relations over sets. We need precise notation and terminology for sets and relations in order to talk about databases. Basic mathematical facts about relations and sets are required to understand how a database is designed and implemented.

Relations

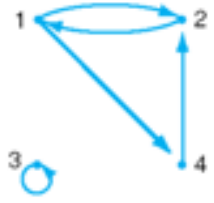
- Definition
- How to represent a binary relation as a directed graph or a Matrix
- Unary relations
- Properties of binary relations
- Transitive closure
- Equivalence relations and partitions
- Partial orders and total orders

Examples

Let the universal set be $\mathbb{Z}$, $A = \{a \in \mathbb{Z} \mid a \text{ is even, and } 2 < a < 8\}$,
$B = \{b \in \mathbb{Z} \mid b \text{ is odd}, b \neq 0, b \neq 5 \text{ and } 2 < b < 8\}$,
$C = \{c \in \mathbb{Z} \mid c \text{ is odd}, 3 < c < 7 \text{ and } 2c = 9\}$ and $D = \{d \in \mathbb{Z} \mid d^2 + 3d + 2 = 0\}$. Find the elements of the following expressions:

1. $(A \cup C) \cap (B - D)$
2. $B \cup (A - C)$
3. $(B \cap \sim C) \cap (A \Delta C)$
4. $\sim(((A \cup C) \cap (B - D)) \cup ((B \cap \sim C) \cap (A \Delta C)))$

Let $A = \{1,2,3,4\}$ and $R = \{(1,4),(1,2),(4,2),(2,1),(3,3)\}$. Represent R over A using a digraph and determine the corresponding adjacency matrix of the digraph.



$$\begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

Functions

- A function is just a map from a set of inputs to a set of outputs. This is exactly what an algorithm computes.
- A function is actually a special kind of relation and functions can be composed (put together) and inverted once we know how to invert and compose relations.

Functions

Functional Programming: A functional programming language manipulates symbolds using basic primitive functions. The power of such languages lies in their ability to build complicated processes from simple ones by composition of the basic primitive functions.

Functions can be used to determine how long algorithms take to run.

Examples

- Let $f: \mathbb{R} \to \mathbb{R}$, check whether $f(y) = \frac{y+7}{3y-4}$ and $f^{-1}(y) = \frac{4y+7}{3y-1}$ are inverses.

- Let $f: \mathbb{R} \to \mathbb{R}$ be given by $f(x) = 5x - 3$. Prove that $f(x)$ is a bijective function.

- Let *A* be a non-empty finite set, and let *f* be a function from $f: A \to A$. Show the following statements are equivalent:
  - *f* is injective,
  - *f* is surjective,
  - *f* is bijective.

Logic and Specification Languages

How can we specify what a program should do? Natural languages can be long-winded and ambiguous and are not appropriate for intricate problems.

A formal language without ambiguous statements is required.

Propositional and Predicate Logic are the most important formal languages for specifying programs.

Propositional Logic

- Syntax: formulas and formal representations
- Semantics: interpretations and truth tables
- Tautologies
- Contradictions
- Semantic consequence
- Logical equivalence

First Order Logic

Whereas propositional logic assumes the world contains facts, first-order logic (like natural language) assumes the world contains

- Objects: people, houses, numbers, colors, baseball games, wars, …
- Relations: red, round, prime, brother of, bigger than, part of, comes between, …
- Functions: father of, best friend, one more than, plus, …

In S2, you will have a module which extends your knowledge of propositional logic, and teaches you predicate logic,
which is a more sophisticated logic in which formulas contain variables that can be quantified.

Examples

- State whether $(a \wedge b)$ and $\sim((a \wedge \sim b) \vee (\sim a \wedge b)) \wedge (a \vee b)$ are logically equivalent. If yes, prove it. If not, explain why.

- Without using any truth table, show that $(P \wedge (Q \vee R)) \rightarrow ((P \wedge Q) \vee (P \wedge R))$ is a tautology.

Examples

Consider the signature $S = \{Cousin, Male, Female, jessie, carol, paul\}$ consisting of a binary predicate symbol *Cousin*, two unary predicate symbols *Male* and *Female*, and three constant symbols *jessie*, *carol and paul*. Assume that these symbols have the following meaning:

- *Cousin* means "is a cousin of" (i.e., *Cousin (a, b)* states *a* is a cousin of *b*).
- *Male* means "is male" (i.e., *Male(a)* states *a* is male).
- *Female* means "is female" (i.e., *Female(a)* states *a* is female).
- *jessie, carol and paul* refer to "Jessie", "Carol" and "Paul", respectively.

Translate the following sentences into *S*-formulae; that is, for each of the following sentences provide an *S*-formula that expresses the sentences:

1. Jessie is a cousin of Carol.
2. Paul has a female cousin.
3. All cousins of Paul are also cousins of Carol.
4. Jessie has at least 2 male cousins.
5. Everybody has a cousin.
6. Nobody is a cousin of everybody else.

Combinatorics

Combinatorics includes the study of counting and also the study of discrete structures such as graphs. It is essential for analysing the efficiency of algorithms.

Combinatorics

- Notation for sums and products, including the factorial function.
- Principles for counting permutations and combinations, for example, to enable you to solve the problem on the following slide.

An example

A small company employs eight people in the manufacturing department, five in the marketing department and three in the accounting department. A project team of six is to be formed to discuss the launch of a new product. In how many ways can the team be formed if:

1. there are exactly two representatives from each department?
2. there are at least two members from the manufacturing department?

Probability and Statistics

- sample space
- events
- discrete probability
- conditional probability
- independence
- random variables
- expectation

Examples

Probability is the likelihood or chance of an event occurring.

$$\text{Probability} = \frac{\text{the number of ways of achieving success}}{\text{the total number of possible outcomes}}$$

For example, the probability of flipping a coin and it being heads is ½, because there is 1 way of getting a head and the total number of possible outcomes is 2 (a head or tail). We write P(heads) = ½ .

- The probability of something which is certain to happen is 1.
- The probability of something which is impossible to happen is 0.
- The probability of something not happening is 1 minus the probability that it will happen.

Other examples

1. If you draw a card from a standard deck of cards, what is the probability of drawing a face card?

2. There are 6 balls in a bag, 3 are red, 2 are yellow and 1 is blue. What is the probability of picking a yellow ball?

Please feel free to ask questions

&

Ka Lok Man - Office hours:
1 pm – 2 pm every Monday and Friday