COMP9020

Foundations of Computer Science
Term 3, 2024

Lecture 18: Course Review

Outline

Final Exam - preparation

Final exam - on the day

Lecture contents - review

Final Exam - preparation

Goal: to check whether you are a competent computer scientist.

Requires you to demonstrate:

- understanding of mathematical concepts
- ability to apply these concepts and explain how they work

Lectures, tutorials, quizzes and mid-term exam have built you up to this point.

Examiner's comments

The questions are intended to assess your **understanding** and your **ability** rather than your knowledge.

Unless specified, *any* valid (mathematical) proof technique is acceptable.

Partial marks are always available for incomplete answers.

Final exam - revision strategy

- Re-read lecture slides
- **Review/solve tutorial, quizzes and mid-term exam**
- Solve more problems from the textbooks

(Applying mathematical concepts to solve problems is a skill that improves with practice)

Outline

Final exam - contents

There will be :

- 10 Questions, 120 marks total

Each question will roughly correspond to one week of lectures:

1. Number theory
2. Set theory and languages
3. Relations and functions
4. Recursion and induction
5. Propositional logic
6. Combinatorics
7. Probability
8. Graph theory
9. Algorithmic analysis

Final exam - Special consideration

Review the UNSW policy on Special Consideration.

UNSW has a “fit-to-sit” policy: by undertaking the assessment you are declaring that you are fit to do so.

If there are any foreseeable issues you must apply before the exam. If circumstances prevent you from applying before the exam, you must apply as soon as possible (within 3 days).

Take Notice

Supplementary exams are only available to students granted Special Consideration.

Outline

Week 1 - number theory

Recap

- Number theory
  - Floor $\lfloor \cdot \rfloor$ and ceiling $\lceil \cdot \rceil$ functions
  - Absolute value $| \cdot |$ function
  - GCD, LCM

Techniques

- How to present proofs

Week 2 - Formal languages and set theory

Recap

- Symbols, words, languages
- Language definitions: Σ^*, $\mathrm{length}()$, concatenation

Techniques

- Definitions around formal languages
- Proofs using Laws of Set Operations

Week 2 - Formal languages and set theory

Recap

- Set notation: $\in$, $\emptyset$, $\mathcal{U}$, $\subseteq$,
- Set operations: $\cap$, $\cup$, ^{c}, $\setminus$, $\mathrm{Pow}()$, $\times$
- Cardinality: $|X| = \#(X)$
- Venn diagrams

Techniques

- How to define sets
- Proofs using the Laws of Set Operations
- Cartesian product
- Cardinality computations
- Proving two sets A and B satisfy $A = B$ either (i) element-wise, (ii) using $A \subseteq B$ and $B \subseteq A$, or (iii) using the Laws of Set Operations

Week 2 - Laws of Set Operations

For all sets A, B, C:

Commutativity	$A \cup B = B \cup A$
	$A \cap B = B \cap A$
Associativity	$(A \cup B) \cup C = A \cup (B \cup C)$
	$(A \cap B) \cap C = A \cap (B \cap C)$
Distribution	$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$
	$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$
Identity	$A \cup \emptyset = A$
	$A \cap \mathcal{U} = A$
Complementation	$A \cup (A^c) = \mathcal{U}$
	$A \cap (A^c) = \emptyset$

Week 2 - Other useful set laws

Idempotence	$A \cap A = A$
	$A \cup A = A$
Double complementation	$(A^c)^c = A$
Annihilation	$A \cap \emptyset = \emptyset$
	$A \cup \mathcal{U} = \mathcal{U}$
de Morgan's Laws	$(A \cap B)^c = A^c \cup B^c$
	$(A \cup B)^c = A^c \cap B^c$

Theorem (Principle of Duality)

If you can prove $A_1 = A_2$ using the Laws of Set Operations then you can prove $dual(A_1) = dual(A_2)$

Theorem (Uniqueness of complement)

$A \cap B = \emptyset$ *and* $A \cup B = \mathcal{U}$ *if, and only if,* $B = A^c$.

Week 3 - Relations

Recap

Relations:

- Definitions, n-ary relations

Binary Relations:

- Representations: Matrix, graphical, directed graph
- Properties: Reflexivity, Antireflexivity, Symmetry, Antisymmetry, Transitivity
- Functions: Domain, codomain, image

Week 3 - Relations

Equivalence Relations:

- Reflexive, Symmetric, Transitive
- Equivalence classes, Partitions

Partial Orders:

- Generalize $\leq$
- Reflexive, Antisymmetric, Transitive
- Hasse diagram, minimum vs minimal, glb/lub, lattice

Recap

Partial Orders:

- Reflexive, Antisymmetric, Transitive
- Hasse diagram, minimum vs minimal, glb/lub, lattice, topological sort, product/lexicographic/lenlex order

Functions:

- Functional composition: $g \circ f = f; g$
- Inverse function: $f^{\leftarrow}$, when it is a function
- Matrices: Add, Scalar multiplication, Matrix multiplication, Transpose
- big-O notation: O, Ω, Θ

Recap

- 2-element Boolean Algebra
- Boolean functions
- CNF/DNF, canonical DNF
- Karnaugh maps, optimal DNF

Recap

- Syntax:
  - Well-formed formulas
  - Parse trees
- Semantics:
  - Truth assignments
  - Truth tables
  - Logical equivalence
  - Entailment

Recap

- Recursive datatypes: Natural numbers, Words, Expressions, Well-formed formulas
- Recursive programming/functions: Factorial, concatenate, length
- Recurrence equations
  - Unwinding
  - Approximating with big-O
  - Master Theorem

Recap

- Basic induction
- Variants of basic induction
- Structural induction

Week 8,9 - Combinatorics, probability, statistics

Recap

Combinatorics

- Basic counting rules: Disjoint sets, Cartesian products
- Permutations and Combinations
- Balls in boxes
- Using recursion to count

Probability

- Sample spaces, probability distributions, events
- Independence
- Recursive probability calculations

Statistics

- Random variables
- Expectation, linearity of expectation
- Variance, standard deviation

Weeks 8,9 - Selecting items summary

Selecting k items from a set of n items:

With replacement	Order matters	Balls in boxes	Formula
Yes	Yes	Distinguishable balls Multiple balls per box	n^k
No	Yes	Distinguishable balls At most one ball per box	$(n)_k$
No	No	Indistguishable balls At most one ball per box	$\binom{n}{k}$
Yes	No	Indistinguishable balls Multiple balls per box	$\left(\!\binom{n}{k}\!\right) = \binom{n+k-1}{k}$

Week 9,10 - Graph theory

Recap

- Definitions and notation: vertices, edges, paths, cycles, connectedness, isomorphisms
- Graph classes: Trees, Complete graphs, complete k-partite graphs
- Graph traversals: DFS/BFS, Eulerian path/circuit, Hamiltonian path/cycle
- Graph properties: Chromatic number, Clique number, Planarity

Recap

- Count "cost" (default: running time) of elementary operations as a function of (a parameter of) the input
- Approximates real-world cost
- Using both big-O and worst-case to simplify analysis
- Recursive algorithms lead to recurrence equations

Good luck everyone!