Module Outline and Exam Revision

COMP526: Efficient Algorithms

January 6, 2025

This note gives an exhaustive list of the topics that may appear in the final exam for *COMP526: Efficient Algorithms*.

Contents

1 Logic, Proof Techniques, and Asymptotic Notation

Definitions and operations to know

- Logical proposition
- Logical connectives $\wedge$, $\vee$, $\neg$, $\implies$, $\iff$
- Truth tables
- Satisfiability, Contradiction, Tautology
- Logical equivalence
- Logical predicate
- Existential and universal quantifiers $\exists$ and $\forall$
- Negation of quantified expressions

Concepts/Techniques Proofs will not be tested explicitly on the exam, but you should be familiar with the following techniques employed throughout the module:

- Direct proof: $P \implies Q$
- Proof by contraposition: $(P \implies Q) \equiv (\neg Q \implies \neg P)$
- Proof by contradiction: $(P \implies Q) \equiv ((P \wedge \neg Q) \implies \text{false})$
- Proof by exhaustion: $(P \implies Q) \equiv ((P \wedge A \implies Q) \wedge (P \wedge \neg A \implies Q))$

- Mathematical induction
- Loop invariants
- Amortized analysis

2 *Machines and Models*

Computational Models

- The RAM model, supported operations and their running times
- The PRAM (Parallel RAM) model

Asymptotic Notation

- Definitions of O, Ω, Θ, ω, and o
- Comparison of classes of asymptotic growth: constant, poly-logarithmic, polynomial, exponential
- How asymptotics interact with arithmetic
- Identifying dominant term(s) in an expression

3 *Fundamental Data Structures*

Abstract Data Types

- Array ADT
- Stack
- Queue
- Priority Queue
- Map/Associative Array/Dictionary
- Set

Data Structures & Implementations

- Array data structure
- Linked List
- Binary Trees
  - Complete Binary Tree
  - Binary search trees
  - Balanced Binary Tree (AVL Tree)
- Heap
- Trie
- Amortized analysis of a sequence of operations

4 *Efficient Sorting*

Elementary Sorting Algorithms

- SelectionSort
- InsertionSort
- BubbleSort

Sorting by Divide & Conquer

- MergeSort
- QuickSort
- RadixSort

Other Sorting Methods and Concepts

- HeapSort
- CountingSort
- Lower bound for comparison based sorting algorithms

Divide & Conquer Beyond Sorting

- Binary search of sorted arrays
- k-selection problem
- Majority problem
- Closest points in the plane

5 *String Matching*

- String matching problem definition and variations (first occurrence, all occurrences)
- Brute force algorithm for string matching
- DFA algorithm for string matching
    - DFA lookup table construction
- Knuth-Morris-Pratt (KMP) algorithm of string matching
    - Failure link automaton
    - Failure link array definition and computation
- Boyer-Moore algorithm

6 *Compression*

- Data compression task definition
- Source text, coded text, encoding, decoding
- Compression ratio
- Lossless vs lossy compression
- Character encoding
- Prefix codes (and their connection to trees)
- Fixed length vs variable length codes
- Huffman codes
  - Optimality of Huffman codes as character codes
  - Huffman tree construction
  - How to apply tie-breaking rules for tree construction
  - Encoding and decoding with the Huffman tree
- Intuitive interpretation of entropy (not formal definition)
- Limitations of general compression
  - Kolmogorov complexity
  - Definition of Kolmogorov complexity
  - Non-computability of Kolmogorov complexity
- Run-length encoding (RLE)/Elias encoding
  - encode/decode text using RLE
- Lempel-Ziv-Welch (LZW) Encoding
  - encode/decode using LZW encoding
- Move-to-Front (MTF) Transform
  - encode/decode using MTF transform
- Burrows-Wheeler Transform
  - apply Burrows-Wheeler transform to a text
  - apply inverse Burrows-Wheeler transform to a text

7 *Error-Correcting Codes*

- Definition of error correction and detection tasks
- Definition of block codes, Hamming distance, code distance
- Decoding block codes
- Lower bounds (requirements) for detecting and correcting using block codes

- Parity bits
- (7, 4) Hamming codes
  - how to encode a message
  - detecting errors in encoded messages
  - correcting errors in encoded message
- How Hamming codes are generalized to larger block lengths

8 Parallel Algorithms

- Understand the PRAM model and processing elements (PEs) conceptually; pseudocode for parallel algorithms ("in parallel" keyword)
- Definitions of span/time/depth and work, and how these quantities can be computed
- Definition of work-efficient algorithm
- Understand Brent's theorem
- Parallel Searching
  - brute-force parallel string matching (span and work)
  - parallel Knuth-Morris-Pratt algorithm (span and work)
- Comparator networks and sorting networks
  - interpretation of a comparator network and execution of comparator networks on an input
  - definition of sorting network
  - definitions of size and depth of a comparator network
  - relationship between simple sorting algorithms and sorting networks (e.g., insertion sort)
- Parallel MergeSort algorithm
  - Parallel merge operation
  - Span and work of Parallel MergeSort

9 Text indexing

- Building and searching a trie data structure for a given pattern
- Compact tries
- Suffix tree definition and computation
  - computation with the "naive" algorithm
  - using suffix trees for string matching
  - using suffix trees for finding repeated substrings
- Suffix array definition and computation
- Longest common prefix array definition and computation
- Inverse suffix array and computation