

# COMPUTER SCIENCE PHD QUALIFYING EXAMINATIONS

## TOPICS AND READING LIST

### Part 1: Data Structures and Algorithms

- Abstract data types
- Asymptotic notation (big-o, little-o, big-theta, big-omega) for analysis of time and space
- Complexity in major search, sort, and graph algorithms
- Analysis of recursive programs
- Algorithm design strategies: backtracking, divide and conquer, greedy, dynamic programming, heuristic
- Sorting algorithms and analysis: selection sort, insertion sort, merge sort, quicksort, radix sort, and heap sort
- Search algorithms and analysis: linear search, binary search tree, B-tree, hashing
- Lists: data representation and basic operations (head, last, merge, concatenation), circularly linked lists, generalized list (list of lists)
- Stacks and queues
- Trees: data representation, general trees, binary trees, tree traversal, decision trees, threaded trees, heaps, balanced trees (AVL and 2-3 trees)
- Priority queues
- Graphs: data representation, depth-first search, breadth-first search, topological sort, minimal cost spanning trees, shortest paths
- Correctness of algorithms

Suggested reading:

- ✓ Mark Weiss, *Data Structures and Algorithm Analysis*. Addison-Wesley, 1995.

## Part 2: Systems

The Systems portion of the Qualifying Exam is made up of Computer Architecture or Operating Systems. You will need to select which section you will take when you submit your Letter of Intent.

### Computer Architecture

- Instruction set design, including pros and cons of design features
- Computer arithmetic including number representations, ALU design, multiplication, division, and floating point
- Processor design: non-pipelined and pipelined, including datapath and control options, performance issues and techniques to diminish their impact, and the basics of superscalar and dynamic pipelining
- Memory hierarchy design, including cache, memory, TLB, and virtual memory design, placement and replacement policies, and related performance issues
- Input/Output including I/O device and system design, I/O performance measures, and interfaces, as well as related performance issues
- Basics of multiprocessor design including bus-connected multiprocessors, network-connected multiprocessors, clusters, and network topologies
- Performance comparison using various performance metrics and application of Amdahl's law

Suggested reading:

- ✓ David Patterson and John Hennessy, *Computer Organization and Design: The Hardware/Software Interface*, 3rd Edition, Morgan Kaufmann, 2004.

## Operating Systems

- Role of OS: Resource manager and extended machine; Function and role of interrupts and traps.
- Processes and threads: models, implementation, scheduling algorithms.
- Communication/isolation: User/supervisor mode; Role of traps in system calls; Pipes and messaging; Shared memory; Race conditions.
- Coordination: Synchronization models, algorithms, and usage (including mutex, counting semaphore); Resource management; Deadlock and approaches to avoiding it; Monitors; Blocking and non-blocking models.
- Memory management: Virtual and physical memory concepts; Paged and segmented memory translation (algorithms, data structures, translation HW, demand loading, fragmentation implications); Interaction of system API and user-mode allocation libraries.
- I/O: Devices: Data-transfer (programmed and DMA); Scheduling models (polled, interrupt driven); Structure of drivers; Characteristics of devices; Abstractions provided to kernel and user programs; Role of buffer cache.
- Persistent storage: Filesystem abstractions and implementations (Inode, FAT, CD-ROM,); Backup strategies; Ownership and permissions.

Suggested reading:

- ✓ Andrew Tannenbaum, *Modern Operating Systems*, 2nd Edition, Prentice Hall, 2001. Chapters 1-6.

## Part 3: Theory of Computation

- Regular expressions, regular languages
- Finite automata, non-deterministic finite automata
- Kleene's theorem
- Context-free languages and grammars
- Pushdown automaton (deterministic and non-deterministic)
- Regular grammars, Chomsky normal forms
- Pumping lemmas and closure properties of RL and CFL
- Turing machines: deterministic, non-deterministic, multi-tape
- Church-Turing thesis
- Decidability
- Reducibility
- Complexity classes: P, NP-complete, NP-hard, PSPACE
- Intractability

Suggested reading:

- ✓ Harry Lewis and Christos Papadimitriou, *Elements of the Theory of Computation*, 2nd Edition, Prentice Hall, 1997.

## Part 4: Design and Implementation of Programming Languages

- Stages of programming language interpretation and compilation, including lexical and syntax analysis: context-free grammar in BNF and EBNF; parsers (recursive descent, LR)
- Programming language semantics: operational semantics, axiomatic semantics, and denotational semantics; attribute grammars to describe static semantics
- Analysis and evaluation of control abstractions of programming languages
- Analysis and evaluation of expressions and assignment statements
- Data types and type checking
- Scopes, naming, and storage bindings
- Analysis of design dimensions of subprograms, including parameter passing methods, subprograms as parameters, and overload subprograms
- Support for object-oriented programming
- Language support for concurrency
- Abstract data types
- Exception and event handling
- Selection of programming paradigm and language based on problem or domain (object-oriented, logic, functional, and imperative)

Suggested reading:

- ✓ Robert W. Sebesta, *Concepts of Programming Languages*, 8th Edition, Addison Wesley, 2007.