

Ph.D. Written Examination Syllabus

April 30, 2008

Architecture Syllabus

1. Quantitative principles of computer design (PAT, Chapters 1 and 2)
 - Technological trends.
 - Quantitative principles of computer design.
 - Instruction set analysis and design.
 - Cost analysis.
2. Instruction pipelining: basic and advanced concepts (PAT, Appendix A and Chapters 3 and 4)
 - Pipeline performance.
 - Pipeline hazards.
 - Control of pipeline stages.
 - Pipelining multicycle operations.
 - Pipeline scheduling and loop unrolling.
 - Dynamic scheduling (Tomasulo's algorithm and scoreboarding).
 - Branch prediction.
 - Advanced pipelining: software pipelining,
 - Superpipelined, superscalar, and VLIW processing.
3. Vector architectures and vectorizing compilers (PAT, Appendix G)
 - VMIPS vector architecture.
 - Vectorization and performance enhancement techniques.
 - Performance analysis.
4. Memory-hierarchy design (PAT, Chapter 5)

- Cache memory design and analysis.
- Techniques for reducing miss rates, miss penalty, hit time.
- Main memory and virtual memory design.
- Memories for pipelined and vector architectures.

5. I/O and Storage Systems (PAT, Chapter 7)

- Buses.
- I/O performance measures.
- Disk arrays.
- Designing I/O systems.

6. Interconnection networks (PAT, Sections 8.1–8.5)

- Shared and switched networks.
- Connection-oriented and connectionless communication.
- Topology and routing.
- Performance issues.

7. Multiprocessors (PAT, Sections 6.1, 6.3, 6.5, and 6.7):

- Centralized and distributed shared memory architectures.
- Basic Cache coherence protocols: snoopy and directory based.
- Synchronization mechanisms in hardware and software.

8. Computer arithmetic (PAT, Appendix H):

- Fast adders and multipliers.
- Floating-point arithmetic.

References

- Patterson and Hennesy, Computer Architecture: A Quantitative Approach, 3rd edition (PAT).

Analysis of Algorithms Syllabus

Philosophy: We are testing analytical ability: how well students think on their feet, rather than details of some data structure, etc. So there will be less reliance on memory. The questions will be more of a general nature (and will require more analytical/creative ability) than a typical final exam.

All sections marked with a * in CLRS (4.4, 5.4, 11.5, 12.4, 16.4, 16.5, 21.4, 26.4, 26.5) are excluded.

1. Preliminaries (CLRS Chapters 3 and 4):

- pages 21, CLRS Ch 3 Growth of Functions
- pages 29, CLRS Ch 4 Recurrences
- pages 30, CLRS Ch 5 Probabilistic Analysis and Randomized Algorithms
- pages 6, CLRS Ch C.3 Discrete random variables

- Algorithm growth, Big-Oh and similar notations.
- Recurrence equations, and their solution. Applications of recurrences.
- Randomized algorithms, expected runtime analysis

2. Sorting (CLRS Chapters 6-9):

- pages 18, CLRS Ch 6 Heapsort
- pages 20, CLRS Ch 7 Quicksort
- pages 18, CLRS Ch 8 Sorting in Linear Time
- pages 13, CLRS Ch 9 Medians and Order Statistics

- Algorithms (Heapsort, quicksort, mergesort, radixsort).
- Proof that sorting (by comparisons) takes $\Omega(n \log(n))$ time.
- Medians and order statistics.

3. Divide and conquer techniques (CLRS Section 2.3, Chapter 7, and Section 28.2):

- pages 10, CLRS Section 2.3 Designing Algorithms
- pages 20, CLRS Ch 7 Quicksort
- pages 7, CLRS Section 28.2 Strassen's algorithm

- For mergesort.
- For quicksort.
- Strassen's algorithm for matrix multiplication.

4. Trees and hashing (CLRS Chapters 10–14 and 18):

- pages 21, CLRS Ch 10 Elementary Data Structures
- pages 42, CLRS Ch 11 Hash Tables
- pages 20, CLRS Ch 12 Binary Search Trees
- pages 29, CLRS Ch 13 Red-Black Trees
- pages 21, CLRS Ch 18 B-trees
- pages 17, CLRS Ch 14 Augmenting Data Structures

- Stacks, queues, linked lists.
- Hashing techniques and functions.
- Binary search trees.
- Balanced search trees.
- B-trees.
- Dynamic order statistics, interval trees

5. Dynamic programming (CLRS Chapter 15):

- pages 47, CLRS Ch 15 Dynamic Programming

- Examples and principles.

6. Greedy algorithms (CLRS Chapters 16 and 23):

- pages 25, CLRS Ch 16 Greedy Algorithms
- pages 19, CLRS Ch 23 Minimum Spanning Trees

- Huffman codes.
- Minimal spanning tree.

7. Graph algorithms (CLRS Chapters 22-26):

- pages 34, CLRS 22 Elementary Graph Algorithms
- pages 19, CLRS 23 Minimum Spanning Trees
- pages 40, CLRS 24 Single-Source Shortest Paths
- pages 23, CLRS 25 All-pairs Shortest Paths
- pages 56, CLRS 26 Maximum Flow

- Graph representations.

- Graph traversals.
- Spanning trees or forests.
- Shortest paths.
- Maximum flow.

8. Amortized Analysis (CLRS Ch 17)

- pages 25, CLRS 17 Amortized Analysis
- Analysing a sequence of operations
- Aggregate, accounting, and potential methods

9. Data Structures for Disjoint Sets (CLRS Ch 21, excluding 21.4)

- pages 12, CLRS 21 (excluding 21.4) Data Structures for Disjoint Sets
- Union/Find data structures
- Linked list representations, disjoint forest representations
- Union by rank, path compression
- Runtime, but not the proof, of union by rank with path compression

10. NP-completeness (CLRS Chapter 34):

- pages 56, CLRS Ch 34 NP-Completeness
- We won't ask students to prove how NP Turing Machines reduce to satisfiability, but they should be able to do simple reductions.
- The classes P, NP, NP-complete, NP-hard.
- An initial NP-complete problem.
- Reductions and proofs that other problems are NP-complete, e.g., Hamiltonian Circuit, Traveling Salesman, and Vertex Cover.

References

- Algorithms, by Cormen, Leiserson, Rivest, and Stein (CLRS) (the "green" book).

Operating Systems Syllabus

1. Basic resource management:

- pages 95-128 SGG Ch 4 Processes
- pages 207-249 CDK Ch 6 Operating Systems Support
- pages 129-149 SGG Ch 5 Threads
- pages 151-188 SGG Ch 6 Process scheduling
- pages 273-316 SGG Ch 9 Memory Management
- pages 317-370 SGG Ch 10 Virtual Memory
- pages 371-409 SGG Ch 11 File System Interface
- pages 411-451 SGG Ch 12 File System Implementation

Total: 314 pages

Main Topics:

- Processes and process organization
- Threads versus processes
- Process scheduling
- Memory management and virtual memory
- Disk and storage management
- File systems

2. Synchronization:

- pages 189-241 SGG Ch 7 Process Synchronization
- pages 243-270 SGG Ch 8 Deadlocks

Total pages: 82 pages

Main Topics:

- Mutual exclusion
- Mutex locks
- Semaphores
- Condition variables
- Classical synchronization problems
- Monitors
- Concurrency
- Deadlocks

3. Communication:

- pages 65-124, CDK Ch 3 Networking
- pages 125-164, CDK Ch 4 Interprocess Communication
- pages 165-203, CDK Ch 5 Distributed Objects and Remote Invocation
- pages 669-695, CDK Ch 17 Corba

Total Pages: 166

Main Topics:

- Connectionless versus connection-oriented protocols
- Blocking versus non-blocking
- Client-server model
- RPCs
- Remote object invocation
- CORBA
- Multicast
- Group communication
- Message passing

4. Naming, authentication, and protection

- pages 251-308, CDK Ch 7 Security
- pages 629-656, SGG Ch 18 Protection
- pages 353-384, CDK Ch 9 Name Services

Total: 118 pages

Main Topics:

- Basic security principles
- Authentication
- Digital signatures
- Public Key versus Shared Key
- Needham-Schroeder, Kerberos, SSL
- Domains
- Capabilities
- Name services and spaces
- DNS
- Directory services

- Discovery services

5. Distributed file systems

- pages 309-352, CDK CH 8 Distributed File Systems

Total: 44 pages

Main Topics:

- Design principles of distributed file systems
- NFS
- Andrew
- Caching
- Consistency
- Persistent stores

Operating Systems total: 724 pages

References

- Operating System Concepts, 6th ed by Silberschatz, Galvin and Gagne (SGG)
- Distributed Systems, 3rd ed. by Coulouris, Dollimore, and Kindberg (CDK)

Programming Languages and Compilers Syllabus

1. Overview of Compilation

- Scott §1.4 Compilation and Interpretation (8 pages)
- Cooper §1.4 Compiler Structure (3 pages)
- Cooper §1.5 High-Level View of Translation (15 pages)
- compilation and interpretation
- structure and phases of a compiler

2. Programming Language Syntax

- Cooper §2 Scanning (46 pages)
- Cooper §3 Parsing (77 pages)
- language classes (regular, LL, LR, context free)
- regular expressions
- finite automata, table driven scanners
- BNF and ambiguous grammars
- top-down vs. bottom up parsing
- recursive descent and table-driven parsers

3. Names, Scopes, and Activation Records

- Cooper §6 The Procedure Abstraction (56 pages)
- Cooper §7.2 Assigning Storage Locations (4 pages)
- Scott §3.5 The Binding of Reference Environments (6 pages)
- variable lifetimes
- storage management
- static and nested scope
- parameter passing
- subroutine closures and static links

4. Semantic Analysis

- Cooper §5.3.1 Syntax-Related Trees (5 pages)
- Cooper §5.7 Symbol Tables (11 pages)
- Scott §4 Semantic Analysis (34 pages)
- abstract syntax trees
- symbol tables

- attribute grammars
- inherited and synthesized attributes

5. Data Types

- Scott §7 Data Types (100 pages)
- Cooper §4.2 (16 pages), §4.5 (2 pages)
- type systems
- type checking
- records/variants, strings, and lists
- arrays, array memory layout
- pointers, recursive types, and garbage collection
- equality and assignment

6. Control Flow

- Scott §6 Control Flow (74 pages)
- Cooper §7.8 Control-Flow Constructs (13 pages)
- Cooper §7.9 Procedure Calls (5 pages)
- order of evaluation within expressions
- unstructured control flow
- structured control flow
- recursion

7. Data Abstraction and Object Orientation

- Scott §9.2 Encapsulation and Inheritance (8 pages)
- Scott §9.4 Dynamic Method Binding (14 pages)
- encapsulation
- inheritance
- dynamic method binding

8. Functional Programming

- Scott §10.2 Functional Programming Concepts (2 pages)
- Scott §10.5 Higher-Order Functions (4 pages)
- functional programming concepts
- higher-order functions
- currying

9. Program Analysis

- Cooper §5.3.2 Graphs (4 pages)
- Cooper §9.2 Iterative Data-Flow Analysis (19 pages)
- Cooper §9.3 Static Single-Assignment Form (25 pages)
- iterative data-flow analysis
- static single-assignment form
- control flow graphs

10. Code Generation

- Cooper §7.3 Arithmetic Operators (9 pages)
- Cooper §7.4 Boolean and Relational Operators (11 pages)
- Cooper §7.8 Control-Flow Constructs (13 pages)
- Cooper §7.9 Procedure Calls (5 pages)
- expressions
- boolean relations
- control constructs
- procedure calls

11. Machine-Independent Optimizations

- Cooper §8.4 Scope of Optimization (4 pages)
- Cooper §8.5 Value Numbering Over Regions Larger Than Basic Blocks (9 pages)
- Cooper §8.6 Global Redundancy Elimination (7 pages)
- Cooper §10.3.1 Eliminating Useless and Unreachable Code (7 pages)
- local vs. global optimization
- dead/useless code elimination
- SSA-based value numbering
- global redundancy elimination

12. Machine-Dependent Optimizations

- Cooper §11 Instruction Selection (37 pages)
- Cooper §12 Instruction Scheduling (34 pages)
- Cooper §13 Register Allocation (40 pages)
- instruction selection, peephole optimization
- instruction scheduling, list scheduling

- register allocation, graph coloring

References

- Programming Language Pragmatics, 2nd Edition, by Michael Scott [Scott]
- Engineering a Compiler by Keith Cooper and Linda Torczon [Cooper]

All “CD” and “Advanced Topics” sections may be omitted from the chapters and sections listed above.