Ph.D. Qualifying Reading Lists

CS 6352 – Performance of Computer Systems and Networks

List of Topics:
• Properties of Poisson streams of customer arrivals.
• Analysis and performance figures of the M/M/1 queue.
• Continuous parameter Markov chains.
• Single state dependent (continuous time) Markovian queueing systems.
• Various applications of such state dependent cases in computer systems and data communication networks.
• Generalized Little's result for multiple non-FIFO queues.
• Development and analysis of Markov chains for simple priority queues.
• Developments of Pollaczek-Khinchin mean value formula for the M/G/1 queue.
• Applications.
• Development of discrete parameter Markov chains for discrete time queues.
• Analysis of discrete parameter Markov chains.
• Evaluation of performance figures.
• Applications of discrete time queues in computer systems and data networks (such as, for examples, cross-bar and simple multistage switches).
• Product form solutions for networks of continuous time open and closed Markovian queues (unlimited buffer, state independent service rates).
• Convolution algorithm and Mean Value Analysis techniques for such closed queuing networks.

Type of questions:
Questions will be combinations of theoretical development, analysis of given systems, development of appropriate models and follow up analysis starting from verbal descriptions of physical systems. In most cases, students should attempt to solve problems from fundamental principles rather than trying to remember and apply formulae for various special cases.

A set of helpful formulae, etc. (such as the Pollaczec-Khinchin mean value formula and the MVA algorithm) will be supplied along with the question paper.

The following list of references include the commonly used text book, other reference books on queues, and a sample of books on Probability Theory. Students are responsible for correcting errors in the reference material.

Text Book:

Other References:
L. Kleinrock, Queueing Systems, Volume 1, Theory. Wiley, 1975

Last Update April 10, 2014
Ph.D. Qualifying Reading Lists

CS 6353 – Compiler Construction
List of Topics:

- Basic automata theory
  - Classification of grammars and languages.
- Lexical analysis
  - Regular expressions, Regular languages.
- Syntax analysis
  - Context free grammars.
  - Top-down parsing techniques: Recursive descent, LL(1).
  - Bottom-up parsing techniques: LR parsing.
- Semantic analysis
  - Synthesized attributes and inherited attributes.
  - Syntax-directed translation.
  - Type checking.
- Code generation
  - Runtime storage management.
  - Backpatching, peephole optimization.
  - Register Allocation: Graph coloring.
- Optimizing techniques:
  - Concepts of basic blocks, loops.
  - Data flow analysis: Framework.

Textbook:
"Compilers: principles, techniques and tools".
CS 6354 – Advanced Software Engineering

Overall References:

1. Overview of Software Engineering:
* Review of technical and management aspects of software engineering: What is software engineering, different roles in a software engineering project, organization of programming teams, management and technical tools.
+ Chapter 1 of Bruegge/Dutoit
+ Chapters 1 and 2 of Sommerville.

2. Software Project Management, Organization, and Communication Issues
* Management concepts; management activities; modes and mechanisms of project communication; project communication activities; rationale management concepts and activities; configuration management concepts and activities.
+ Chapters 3, 8, 10, and 11 of Bruegge/Dutoit
+ Chapters 3, 28, and 33 from Sommerville.

3. Software Life-Cycle Models:
* Software life-cycle models (linear, waterfall, spiral, transformational), identification of applications that require each of these life-cycles.
+ Chapter 12 of Bruegge/Dutoit

4. Software Requirements Specification
* Functional and non-functional requirements, type of faults that occur in requirements specifications and corresponding methods of identifying them, template for writing requirements specification, description of a requirements specification standard.
+ Chapters 4 and 5 of Bruegge/Dutoit
+ Chapters 4, 5, 6, and 7 from Sommerville.

5. Software Architecture:
* Rationale for software architecture, classes of software architecture, examples to illustrate each class, detailed example to demonstrate the role of software architecture in achieving high-quality software.
+ Software Architecture, M. Shaw and D. Garlan, Prentice-Hall.
+ Chapter 13 from Sommerville.

6. Software Design:
* Design quality criteria, top-down/iterative design process, assessment of design quality; detailed case-study to illustrate different designs for an application; metrics for assessing designs.
+ Chapters 6 and 7 of Bruegge and Dutoit
+ Chapters 12, 14, 15, 16, and 17 from Sommerville.

7. Object-Oriented Modeling and Specification:
* Object-oriented programs: Difference between abstract data
types and objects, single and multiple inheritance, example of applications that benefit from object-oriented design, behavioral classes, implementation of objects to support concurrent threads. Modeling with UML.
+ Chapter 2 of Bruegge/Dutoit

8. Software Testing:
* Structural testing (statement, decision, condition, decision/condition, multiple condition, path coverage criteria), Functional testing (equivalence partitioning, boundary value testing), integration testing (description and comparison of bottom-up, top-down and thread integration), static program analysis, symbolic execution, automated test data generation.
+ Chapter 9 of Bruegge/Dutoit
+ Chapters 22 and 23 from Sommerville.

+ Suggest Ghezzi, parts of chapter 5

• Algebraic Specification:
* Abstract data type components: O- and V-functions, constructor and destructor operations, state machine model, algebraic specification (syntax and semantic specification), simple specifications (rules that specify the effect of each O-function on each V-function), specification of exceptions, more complicated specifications (effect of constructor operations on destructor operations), completeness of specification. Systematic development of abstract data type components (identification of data structure, specification of representation invariant, mapping function for V-functions, implementation of V-functions, implementation of initialization operation, implementation of other O-functions, proof using structural induction).
• logic based notation e.g., Z
• graphical notation e.g. Statecharts

10. Formal Verification
+Ghezzi parts of chapter 6
6.4.2 proof of correctness
6.5 symbolic execution
6.6 model checking sections
+ introduction to theorem provers
+ tutorial papers on model checking, theorem provers

• Weakest Pre-Conditions:
Ph.D. Qualifying Reading Lists

* Definition, rationale, derivation of weakest preconditions for assignment, concurrent assignment, conditional, and iterative statements, loop invariants, simple program verification.
CS 6360 – Database Design

Textbook: "Fundamentals of database systems"
by Elmasri and Navathe:

Topics
-------

Database System Concepts and Architecture (Data models, Schemas, Instances, Database architecture, classification)

Entity-Relationship (ER) model, ER diagrams

The Enhanced Entity-Relationship (EER) model, EER Diagrams

Relational Data Model, Relational algebra, SQL

Relational Database Design by ER/EER-to-Relational Mapping

Database Design Theory and Normalization (Basics of Functional Dependencies and Normalization for Relational Databases; Algorithms for Relational Database Schema design)

Query processing and optimization

Transaction processing concepts and theory

Concurrency Control Techniques

Database Recovery Techniques

Last Update April 10, 2014
Ph.D. Qualifying Reading Lists

CS 6361 – Requirements Engineering

Topics:
Requirements Engineering: Introduction
Why RE? – error propagation in software lifecycle, cost and size
of requirements errors, aims and scope
What is RE? – What are requirements? Role of requirements and
requirements engineers
How to do RE? – types of errors, formal vs. semi-formal vs.
informal
Requirements Engineering Processes
Why process? A framework for RE process: Elicitation,
specification, validation, RE process and software lifecycle
models
Requirements Elicitation: Part I
Why is it difficult? What to elicit? How to elicit?
Requirements Elicitation: Part II
Advanced goal-directed strategy, knowledge acquisition,
data/information elicitation techniques
Scenario Analysis
Use cases, episodes, scripts, cycle of natural inquiry, abstract
vs. concrete scenario, scenario space
Requirements Analysis, Modelling and Specification: Review
Conceptual modeling perspective of basic RE process, carving the
product space
Object-Oriented Modeling:
Intellectual origins, conceptual modeling, UML overview
Enterprise Requirements & Functional Requirements: Structural
Requirements
Agent-oriented approach to enterprise modeling, ERD, i*, JSD,
SADT, IDEF
Functional Requirements: A Formal OO-RML/Telos epistemological
primitives, ontological primitives, interval calculus,
axiomatization of OO
Functional Requirements: Behavioral Requirements
Decision-oriented behavioral models, State-oriented Behavioral
models (Finite State Machines, StateCharts, PetriNets),
Function-oriented behavioral models.
Non-Functional Requirements
Types of NFRs, classification schemes, Process-oriented approach,
Product-oriented approach, Portability, Reliability, Efficiency,
Usability, Security
Main reference: Lecture Notes at
http://www.utdallas.edu/~chung/RE/contents.html
Additional References:
Articles:
Axel van Lamsweerde,
"Requirements engineering in the year 00: a research perspective",
Proc., Int. Conf. on Software Engineering (ICSE) 2000, pp. 5-19.
Axel Van Lamsweerde", "Goal-Oriented Requirements Engineering: A Guided
Tour",
Proc., Int. Symposium on Requirements Engineering (RE’01), pp.249-261.
Sol Greenspan, John Mylopoulos, Alex Borgida, "On Formal Requirements
Modeling
Languages: RML Revisited", Proc., Int. Conf. on Software Engineering (ICSE)
1994, pp. 135-147.
Mike Wooldridge and Nick Jennings, "Software Engineering with Agents:
Fitsfalls
URL = http://www.jfipa.org/publications/AgentOrientedSoftwareEngineering/

Anton, A.I.; Potts, C. "The use of goals to surface requirements for evolving systems", Proc., Int. Conf. on Software Engineering (ICSE) 1998, pp. 157 -166


Books:
------

A. M. Davis, Software Requirements: Objects, Functions, & States

P. Loucopoulos and V. Karakostas, System Requirements Engineering,

M. Jackson and T. DeMarco, Software Requirements and Specifications,
Addison-Wesley, 1995.

R. H. Thayer and M. Dortman, Software Requirements Engineering: 2nd edition,

I. Sommerville and P. Sawyer, Requirements Engineering - A Good Practice


J. Martin and J. Odell, Object-Oriented Methods: A Foundation, Prentice-Hall,
1995.

Reference

L. Chung, B. Nixon, E. Yu and J. Mylopoulos, Non-Functional Requirements in
Ph.D. Qualifying Reading Lists

CS 6362 – Software Architecture and Design

Topics:

Introduction to Software Architecture Classical

Module Interconnection

Languages Abstract DataTypes and Objects Module

Decomposition Issues

Data Flow

Repositories Events

Process Control

JavaBeans

Client Server

Middleware: CORBA, OLE/DCOM, J2EE/J2ME, .Net Patterns


Articles:

• David Kalinsky, “Design Patterns for High Availability”, March 13, 2003

URL: http://www.eetimes.com/story/OEG20020729S0030

Last Update April 10, 2014
Ph.D. Qualifying Reading Lists


Representation

• P. Clements, Comparing the SEI’s Views and Beyond Approach for Documenting Software Architectures’ with ANSI-IEEE 1471-2000
• Pengcheng Zhang, Henry Muccini and Bixin Li, “A classification and comparison of model checking software architecture techniques”, 2009.

Books:

4) Eric Gamma, Richard Helm, Ralph Johnson and John Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Eric Gamma, Richard Helm, Ralph Johnson and John Vlissides, Addison-Wesley, 1994.

Last Update April 10, 2014


Ph.D. Qualifying Reading Lists

CS 6363 – Design and Analysis of Computer Algorithms
The exam will test knowledge of:
1. major techniques for algorithm design (as listed below);
2. methods to prove algorithm correctness and to analyze its running time;
3. Basic knowledge of NP-Completeness.
NOTE: You should know more than just the algorithms; you are responsible for proving correctness, including all necessary supporting lemmas, and are responsible for proving the correctness of any statements about the asymptotic running times. In addition, you should know the stated subject matter well enough to enable you to provide solutions for closely related questions. Most topics (and knowledge) required are in the CS6363 textbook:
Introduction to algorithms, Second edition, Cormen, Leiserson, Rivest and Stein.

General topics:
Introduction, recurrences and Master Theorem (Theorem 4.1, the proof is not required)
Divide-and-Conquer algorithms
Linear time median selection algorithm (Section 9.3, pp. 189-192)
Closest pair of points in the plane (Section 33.4, pp. 957-961)
Permutation networks (Problem 27-3, page 722)
Sorting Networks (Chapter 27)
Note: students should be able to design divide-and-conquer algorithms for various problems beside those mentioned above
Dynamic Programming
Matrix Chain Order (Section 15.2, pp. 331-338)
Longest Common Subsequence Algorithm (Section 15.4, pp. 350-355.)
All pairs shortest paths (Section 25.2, pp. 629-634)
0/1-knapsack problem (Problem 16.2-2, page 384)
Greedy Method
Huffman's code algorithm (Section 16.3, pp. 385-392)
Minimum spanning tree (Chapter 23)
Single Source Shortest Paths (e.g. Dijkstra's algorithm) (Chapter 24, up to page 601)
Maximum flow (Chapter 26, up to page 668)
Graph algorithms (Chapter 22)
NP-Completeness (Chapter 34, specifically 3SAT, VERTEX COVER, INDEPENDENT SET,
CLIQUE, 3COLOR, HAMILTON CIRCUIT (both directed and undirected), as well as definitions and properties of polynomial time reducibilities.)
Linear programming: (Chapter 29, pp. 770-789 and pp. 804-807.)
CS 6364 – Artificial Intelligence

Text:

Problem solving by search:
Uninformed (Blind) Search and Heuristic (Informed) Search
Problem formulation; Uninformed search strategies: Depth-First Search, Breadth-First Search, Ununiform-Cost Search, Iterative-Deepening.
Informed Search strategies: Greeedy Best-First Search, A*, IDA*.
Heuristic Functions: heuristic domination, inventing admissible heuristics.

Adversary Search (Game Trees)
How to design computer programs that play games intelligently. The MIN/MAX and the ALPHA/BETA-Pruning algorithms, their complexity and efficient implementations.

Knowledge Representation
Propositional logic. Syntax, semantics and inference in prepositional logic as well as reasoning patterns. First Order Logic: syntax and semantics. Resolution in FOL.

Probabilistic reasoning

Bayesian Networks / Belief Networks
Representation of knowledge in uncertain domains. Semantics of Bayesian Networks. Exact inference in Baysian Networks: inference by enumeration; PolyTree Bayesian networks..
Ph.D. Qualifying Reading Lists

CS 6367 – Software Testing, Validation, and Verification
Part 1: Requirements-Based Testing, Inspections
Introduction, Approaches to Reliability, Requirements-based
Testing strategies (Equivalence Partitioning, Boundary value
Analysis, Cause-Effect graphing), Valid and Reliable testing
strategies and the Fundamental Theorem of testing,
the Partition Testing Model, Random/Statistical testing.
Software Inspections and related approaches.
Textbook: Ch 1, 3, 5-8
Wheeler, Brykczynski, Meeson, "Software Inspection: An Industry
Best Practice", IEEE Computer Society Press.
Part 2: Program Proofs
Predicate calculus, validity, theoretical limitations,
deduction systems, the Resolution method. Verification
of Programs (Flowchart Programs, Inductive Assertions,
Termination, Programs with Arrays, extensions).
Chapter 2: Predicate Calculus
Chapter 3: Verification of Programs
Part 3: Structural, Fault-Based Testing Strategies
Structural Testing, Statement, Branch, Predicate, Base-Path,
Path Testing, Variations of Path Testing, Data-Flow Testing,
Domain Testing, Mutation Analysis, other methods. Evaluations
of testing strategies, inclusion, test set size. Integration
testing; Object-oriented Testing
Textbook: Ch. 9-11, 13, 16-20
DeMillo, Lipton, Sayward, "Hints on Test Data Selection: Help
Musa, "Operational Profiles in Software Reliability Engineering",
Ntafos, "A Comparison of Some Structural Testing Strategies",
White, Cohen, "A domain strategy for Computer Program Testing",
Part 4: Reliability Estimation
Failure rate estimation from test outcomes, error-seeding,
reliability growth models.
Notes on Reserve in Library
References:
Lyu: Handbook of Software Reliability Engineering, IEEE Computer
Musa: Software Reliability Engineering, McGraw-Hill.
CS 6371 – Advanced Programming Languages

Topics:
Programming with Functions; Lambda Calculus and ML programming;
Logic programming; Unification and backtracking; Search tree; Programming in Prolog;
Abstract Syntax; Definite Clause Grammars; Grammar Classifications;
Sets, functions, domains; Domain Theory: Primitive and Compound Domains;
Denotational Definition of Programming Languages; Semantics of Imperative Languages; Recursive Functions; Monotonicity, Continuity, and Fix-points;
Introduction to semantics of Logic Programming Languages,
Verification of Programs, Partial Evaluation; Interpretation and Automatic Compilation;
Axiomatic Semantics: Hoare's Axiomatization of partial correctness

References:
Denotational Semantics by D.A. Schmidt.
Elements of ML Programming, Jeffrey D. Ullman, ML97 Edition
Also see the following web page for more details:
http://www.utdallas.edu/~gupta/courses/apl/
CS 6375 – Machine Learning (Syllabus updated Oct 2006)

Topics: Decision Tree Learning, Artificial Neural Networks, Evaluating Hypotheses, Bayesian Learning, Computational Learning Theory, Instance-Based Learning, Markov Decision Processes, Reinforcement Learning, Support Vector Machines, Bagging, Boosting, Hidden Markov Models, and Clustering.

References:
Ph.D. Qualifying Reading Lists

CS 6378 – Advanced Operating Systems (Material in red with strikethrough is no longer in the syllabus)

Clocks and Event Ordering
1. Lamport – Time, Clocks and the Ordering of Events in a Distributed System (1978)

Causal Message Ordering
3. Raynal, Schiper & Toueg – The causal ordering abstraction and a simple way to implement it (1991)

Consistent Global Snapshots

Termination Detection

Distributed Mutual Exclusion

Clock Synchronization

Agreement Protocols

Fault Tolerance and Data Consistency
14. Bernstein, Hadzilacos & Goodman – Distributed Recovery

File System
17. DeCandia et al. - Dynamo: Amazon's highly available key-value store (2007)
CS 6385 — Algorithmic Aspects of Telecommunication Networks
Chapters 2, 3, 5, 6, 7, 10 from
AND
Chapters 1-4 from
Thomas G. Robertazzi, "Planning Telecommunication Networks",
Ph.D. Qualifying Reading Lists

CS 6388 – Software Project Planning and Management

Management Functions
Planning
Defining the Software Process
  _W. Humphrey, Managing The Software Process, Addison Wesley, 1990, Chapter 13
Organizing and Staffing the Project Office and Team
  _Kerzner, Chapter 4
  _COCOMO
Network Scheduling Techniques
  _Kerzner, Chapter 12
Risk Management
  _Elaine Hall, Managing Risk, Addison Wesley, 1998
Pricing and Estimating
  _International Function Point Users Group web site: (http://www.ifpug.org.ifpug)
Software Quality Assurance
Software Configuration Management
The SEI Capability Maturity Model
CS 6390 – Advanced Computer Networks

General topics:
(1) Transport and Routing (including multicasting) protocols,
(2) Quality of service and Weighted Fair Queuing
(3) Mobile IP/Wireless Data,
(4) IPv6,
(5) MPLS,
(6) Peer-to-peer applications.
(7) Voice over IP

Reading List

NOTE: You can find most of these papers at: http://www.utdallas.edu/~ktekarac/courses/Papers/
The paper may also be found in the IEEE/IEE Xplore database and in the ACM Digital library, available from UTD’s library webpage http://www.utdallas.edu/library/collections/journals.htm

Last Update April 10, 2014