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:
T. G. Robertazzi, Computer Networks and Systems: Queueing Theory and Performance Evaluation. Springer, 2000.

Other References:
D. Gross and C. M. Harris, Fundamentals of Queueing Theory. Wiley, 1997

L. Kleinrock, Queueing Systems, Volume 1, Theory. Wiley, 1975

J. J. Higgins and S. Keller-McNulty, Concepts in Probability and Stochastic Modeling. Duxbury Press, 1995

K.S. Trivedi, Probability and Statistics with reliability, Queueing, and Computer Science Applications. First or Second Edition (2001, Wiley)

C. M. Grinstead and J. L. Snell, Introduction to Probability. American Mathematical Society, 1997

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".
A. Aho, R. Sethi and J. Ullman, Addison-Wesley Publishing Company.
Chapters 1-10.

Ph.D. Qualifying Reading Lists

CS 6354 – Advanced Software Engineering

Overall References:

+ Software Engineering, I. Sommerville, Ed. 5, Addison-Wesley, 1996.
+ Object-Oriented Software Engineering, B. Bruegge and A.H. Dutoit, Prentice Hall, 2000.

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.
+ IEEE Standard for Software Project Management Plans, IEEE Computer Society, New York, 1993.

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
+ IEEE Standard for Developing Software Life Cycle Processes, IEEE Computer Society, New York, 1995.

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.
+ D. Garlan, "Research directions in software architecture," ACM Comp. Surveys, Vol. 27, No. 2, June 1995, pp. 257-261.
+ 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.

+ Parnas, D. L., "On the criteria to be used in decomposing systems into modules," Comm. of the ACM, Vol. 15, No. 12, Dec. 1972, pp. 1053-1058.
+ Parnas, D. L., Siewiorek, D. P., "Use of the concept of transparency in the design of hierarchically structured systems," Comm. of the ACM, Vol. 18, No. 7, July 1975, pp. 401-408.
+ 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.
+ Adrion, W. R., Brandstad, M. A., Cheriavsky, J. C., "Validation, verification, and testing of computer software," ACM Computing Surveys, Vol. 14, No. 2, June 1982, pp. 159-192.
+ Chapter 9 of Bruegge/Dutoit
+ IEEE Standard for Test Documentation, IEEE Standards Board, Mar. 1991.
+ Chapters 22 and 23 from Sommerville.

9. Formal Specification

+ 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).
+ Link to Ian Sommerville's chapter on "Algebraic Specification": http://www.comp.lancs.ac.uk/computing/resources/SE6/IG/algebraicspec.pdf
+ Liskov, B.H., Zilles, S.N., "Specification techniques for data abstraction," IEEE Trans. Soft. Eng., Vol. SE-1, No. 1, March 1975, pp. 7-18.
+ Guttag, J., "Abstract data types and the development of data structures," Comm. of the ACM, Vol. 20, No. 6, June 1977, pp. 396-404.
+ Guttag, J., "Notes on type abstraction (version 2)," IEEE Trans. Soft. Eng., Vol. SE-6, No. 1, Jan. 1980, pp. 13-23.
+ Kapur, D., Srivas, M., "Computability and implementability issues in abstract data types," IJCP: 1988, pp. 33-63.

• 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.
+ Dijkstra, E.W., "Guarded commands, nondeterminacy and formal derivation of programs," Comm. of the ACM, Vol. 18, No. 8, Aug. 1975, pp. 453-457.
+ Robinson, L., Levitt, K.N., "Proof techniques for hierarchically structured programs," Comm. Of the ACM, Vol. 20, No. 4, Apr. 1977, pp. 271-283.

Ph.D. Qualifying Reading Lists

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

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 natual 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:
Intelletual 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 Requiremens 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: Pitfalls
and Pratfalls," IEEE Internet Computing, 3(3):20-27, May/June 1999.

Ph.D. Qualifying Reading Lists

A. Tveit, "A Survey of Agent-Oriented Software Engineering",
Proc. of the First NTNU CSGS Conference, May, 2001
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
J.M. Wing, "A Specifier's Introduction to Formal Methods," IEEE Computer, 23(9):8-24, September 1990.
J.A. Hall, "Seven Myths of Formal Methods," IEEE Software, 7(5):11-19, September 1990.
J.P. Bowen and M.G. Hinchey, "Seven More Myths of Formal Methods," IEEE Software, 12(4):34-41, July 1995.

Books:

A. M. Davis, Software Requirements: Objects, Functions, & States Prentice Hall: Englewood Cliffs, 1993.
P. Loucopoulos and V. Karakostas, System Requirements Engineering, McGraw-Hill, 1995.
M. Jackson and T. DeMarco, Software Requirements and Specifications, Addison-Wesley, 1995.
R. H. Thayer and M. Dortman, Software Requirements Engineering: 2nd edition, IEEE Computer Society Press, 1998.
I. Sommerville and P. Sawyer, Requirements Engineering - A Good Practice Guide, Wiley, 1997.
D. Gause and G. Weinberg, Exploring Requirements, Dorset House, 1989.
J. Martin and J. Odell, Object-Oriented Methods: A Foundation, Prentice-Hall, 1995.
J. Rumbaugh, I. Jacobson and G. Booch, The Unified Modeling Language Reference Manual, Addison-Wesley, 1998.
L. Chung, B. Nixon, E. Yu and J. Mylopoulos, Non-Functional Requirements in Software Engineering, Kluwer Academic Publishing, 2000.

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

Main reference: Lecture Notes at http://www.utdallas.edu/~chung/SA/contents.html

Articles:

- Mary Shaw, Paul Clements, "The Golden Age of Software Architecture," IEEE Software, Vol.23, no. 2, pp. 31-39, Mar./Apr.2006
- Paul Clements, Mary Shaw, ""The Golden Age of Software Architecture Revisited, " IEEE Software, vol. 26, no. 4, pp. 70-72, July/August, 2009.
- Hofmeister, C., Kruchten, P., Nord, R., Obbink, H., Ran, A., & America, P. (2207). A General Model of Software Architecture Design derived from Five Industrial Approaches, Journal of Systems & Software, 80(1), 106-126.
- Hassan Gomma, "Advances in Software Design Methods for Concurrent, Real-Time and Distributed Application," in proceedings The Third International Conference on Software Engineering Advances, 2008, pp. 451-456.
- Kendall Scott, The Unified Process Explained. ISBN 0-201-74204-7, 2002, best practices in Architecture and Design.
- Advanced Design Patterns. Re-use
  M-A. Laverdiere; A. Mourad; A. Hanna; M. Debbabi; "Security Design Patterns: Survey and Evaluation", in Canadian Conference on Electrical and Computer Engineering, May 2006, pp. 1605 – 1608
  David Kalinsky, "Design Patterns for High Availability", March 13, 2003
  URL: http://www.eetimes.com/story/OEG20020729S0030

- Ashraf Armoush, Falk Salewski, Stefan Kowaleski, "Design Pattern Representation for Saftey-Critical Embedded Systems", JSEA
- Glen B. Alleman, "Exception Handling in CORBA Environments, The Late Introduction of Distributed Exception Handling in JAVA TM, CORBA-Based COTS Application Domains", 2000.

Representation

- Nenad Medvidovic. Modeling software architectures in unified modeling language. ACM Transactions on Software Engineering and Methodology, 11(1):2-57, January 2002.
- Nenad Medvidovic and Richard N. Taylor. A classification and comparison framework for software architecture description language. IEEE Transactions on Software Engineering, 26(1):70-93, January 2000.
- P. Clements, Comparing the SEI's Views and Beyond Approach for Documenting Software Architectures' with ANSI-IEEE 1471-2000
- Grady Booch, James Rumbaugh, and Ivar Jacobson, Unified Modeling Language User Guide, (2nd Edition), Advanced UML topics, Sections 3, 5, and 6.
- Mugurel T. Ionita, Henk Obbink and Dieter Hammer, Scenario-Based Architecture Evaluation Methods: An overview, International Conference on Software Engineering 2002 (ICSE'02), Orlando, Florida.
- Hans-Peter Hoffmann, UML 2.0-Based Systems Engineering Using a Model-Driven Development Approach, 2005, available at http://www.stsc.hill.af.mil/crosstalk/2005/11/0511Hoffman.pdf
- Michel, M.M. Gala-Edeen, G.H., "Detecting inconsistencies between software architecture views", in proceedings International Conference on Computer Engineering & Systems, 2009. ICCES 2009. pp. 429-434
- Pengcheng Zhang, Henry Muccini and Bixin Li, "A classification and comparison of model checking software architecture techniques", 2009.
- Jeannette M. Wing and Mandanna Vaziri-Farahani, "A case study in model checking software systems", Science of Computer Programming 28 (1997) pp. 273-299.

Books:

1) Mary Shaw and David Garlan, Software Architecture: Perspectives on an Emerging Discipline, Prentice Hall, 1996.
2) L. Bass, P. Clements & R. Kazman, Software Architecture in Practice, Addison Wesley, 1998.
3) A. W. Brown (Editor), Component-Based Software Engineering, IEEE Computer Society, 1996.
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.
5) Wolfgang Pree, Design Patterns for Object-Oriented Software Development, Addison-Wesley Longman, 1995.

6) I. Singh, B. Stearns, M. Johnson, The Enterprise Team, Designing Enterprise Applications with the J2EE Platform, 2/E, Addison Wesley & Benjamin Cummings, 2002.
7) Randy Otte, Paul Patrick and Mark Roy, Understanding CORBA: The Common Object Request Broker Architecture, Prentice Hall, 1996.
8) Robert Orfali, Dan Harkey and Jeri Edwards, The Essential Client/Server Architecture: Survivor's Guide, John Wiley & Sons, 1995.

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)
Multiplication of large integers (Section 7.1, page 219-223, and Problems 7.2, 7.3, page 250, of "Fundamentals of Algorithms, by Brassard and Bratley, Prentice Hall Publ.)
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. 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.)

Ph.D. Qualifying Reading Lists

CS 6364 – Artificial Intelligence
Text:
S. Russell & P. Norvig. *Artificial Intelligence, A Modern Approach*, Second Edition 2002.

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

Modeling uncertainties with probabilities. Inference using Full Joint Distributions. Bayes' Rule. Naïve-Bayesian 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

Textbook: Software Testing by Paul Jorgensen, 2nd Edition, CRC.
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
Myers: The Art of Software Testing, Wiley.
Goodenough + Gerhart, "Toward a Theory of Test Data Selection", IEEE Trans. on Software Engineering, June 1975.
Hamlet+Taylor, "Partition Testing Does Not Inspire Confidence", IEEE TSE, Dec. 1990.
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).
Manna: Mathematical Theory of Computation, McGraw-Hill.
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 for the Practicing Programmer", IEEE Computer, April 1978.
Musa, "Operational Profiles in Software Reliability Engineering", IEEE Software, March 1993.
Ntafos, "A Comparison of Some Structural Testing Strategies", IEEE TSE, June 1988.
White, Cohen, “A domain strategy for Computer Program Testing”, IEEE-TSE, May 1980.
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 Society Press, Mc Graw Hill.
Musa: Software Reliability Engineering, McGraw-Hill.

Ph.D. Qualifying Reading Lists

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
The Art of Prolog, L. Sterling and E. Shapiro. MIT Press, 1997.
Also see the following web page for more details:
http://www.utdallas.edu/~gupta/courses/apl/

Ph.D. Qualifying Reading Lists

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:

Artificial Intelligence (second edition) by Stuart Russell and Peter Norvig, Prentice Hall, 2003.

Machine Learning by Tom Mitchell, McGraw Hill, 1997.

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)
2. Fidge - Logical Time in Distributed Computing Systems (1991)

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

Consistent Global Snapshots
4. Chandy & Lamport - Distributed Snapshots: Determining Global States of Distributed Systems (1985)

Termination Detection
5. Huang - Detecting Termination of Distributed Computations by External Agents (1989)

Distributed Mutual Exclusion
6. Ricart & Agrawala - An Optimal Algorithm for Mutual Exclusion in Computer Networks (1981)
7. Maekawa - A sqrtN Algorithm for Mutual Exclusion in Decentralized Systems (1985)
8. Raymond - A Tree-Based Algorithm for Distributed Mutual Exclusion (1989)

Clock Synchronization
9. Cristian - Probabilistic Clock Synchronization (1989)
10. Gusella & Zatti - The Accuracy of the Clock Synchronization Achieved by TEMPO in Berkeley UNIX 4.3BSD (1989)
11. Mills - Improved Algorithms for Synchronizing Computer Network Clocks (1995)

Agreement Protocols
12. Fischer - The Consensus Problem in Unreliable Distributed Systems (1983)

Fault Tolerance and Data Consistency
13. Koo & Toueg - Checkpointing and Rollback-Recovery for Distributed Systems (1987)
14. Bernstein, Hadzilacos & Goodman - Distributed Recovery

(1987)
15. Jajodia & Mutchler - A Hybrid Replica Control Algorithm Combining Static and Dynamic Voting (1989)

File System
16. Ghemawat, Gobioff & Leung - The Google File System (2003)
17. DeCandia et al. - Dynamo: Amazon's highly available key-value store (2007)

Ph.D. Qualifying Reading Lists

CS 6385 – Algorithmic Aspects of Telecommunication Networks

Chapters 2, 3, 5, 6, 7, 10 from
Robert S. Cahn, "Wide Area Network Design", Morgan Kaufmann, 1998.
AND
Chapters 1-4 from
Thomas G. Robertazzi, "Planning Telecommunication Networks",
IEEE Press, 1999.

Ph.D. Qualifying Reading Lists

CS 6388 – Software Project Planning and Management

Management Functions

_Kerzner. *Project Management: a Systems Approach to Planning, Scheduling and Controlling*. Van Nostrand Reinhold, 1994, Chapters 5 and 3

Planning

_IEEE Std 1058-1998, IEEE Standard for Software Project Management Plans

_Chambers & Associates Pty, Ltd. Concept: Work Breakdown Structure (www.chambers.com.au/Sample_p/wbs_cncp.htm)

Defining the Software Process

_W. Humphrey, *Managing The Software Process*, Addison Wesley, 1990, Chapter 13

_W. Humphrey and M. I. Kellner. "Software Process Modeling: Principles of Entity Process Models". *Proceedings of the 11th International Conference on Software Engineering*, IEEE, 1989

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

_USC COCOMOII Reference Manual, 2000

_International Function Point Users Group web site: (http://www.ifpug.org.ifpug)

Software Quality Assurance

_J. Sanders and E. Curran, *Software Quality: A Framework for Success in Software Development and Support*, Addison Wesley, 1998

Software Configuration Management

_S. J. Ayer and F. S. Patrinostro, *Software Configuration Management: Identification, Accounting, Control and Management*, McGraw Hill, 1992

The SEI Capability Maturity Model

_The Software Engineering Institute, *The Capability Maturity Model*, Addison Wesley, 1995. Chapters 1-5

Ph.D. Qualifying Reading Lists

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
1. Reference book (**Computer Networks** by Peterson and Davie).
2. **Design Philosophy of the DARPA Internet Protocols**, D. Clark, Proc. of ACM SIGCOMM '88.
3. **An Architecture for Wide-Area Multicast Routing**, S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C.-G. Liu, and L. Wei, Proc. of ACM SIGCOMM'94.
4. **Multicast Routing in Datagram Internetworks and Extended LANS**, S. Deering and D. Cheriton, ACM Transactions on Computer Systems, Vol 8 No 2, May 1990, pp. 85-110.
5. **The Stable Paths Problem and Interdomain Routing**, T. Griffin, B. Shepherd, and G. Wilfong, IEEE/ACM Transactions on Networking, Vol 10 No 2, April 2002.
6. **Mobile IP**, C. Perkins, IEEE Communications Magazine, Vol 35, No. 5, May 1997.
7. **Mobility Support in IPV6**, C. Perkins, D. Johnson, ACM Mobicom 1996.
8. **IP Multicast Channels: EXPRESS Support for Large-scale Single-source Applications**, H. Holbrook and D. Cheriton, SIGCOM 1999.
9. **Congestion Avoidance and Control**, V. Jacobson and M. Karels, Proc. ACM SIGCOMM '88.
10. **Random Early Detection Gateways for Congestion Avoidance**, S. Floyd and V. Jacobson, IEEE/ACM Transactions on Networking, Vol. 1, No. 4, pp. 397-413, August 1993.
11. **Equation-Based Congestion Control for Unicast Applications**, S. Floyd, M. Handley, J. Padhye, and J. Widmer, *Proc. of ACM SIGCOMM '00*, Aug. 2000.
12. **Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications**, I Stoica, R Morris, D Liben-Nowell, D R. Karger, M. F Kaashoek, F Dabek, H Balakrishnan, ACM SIGCOMM 2001.
13. **SOS: Secure Overlay Services**, A. Keromytis, V. Misra, and D. Rubenstein (Columbia University), ACM SIGCOMM, Pittsburgh, PA, USA, August 2002.
14. **SIFF: A Stateless Internet Flow Filter to Mitigate DDoS Flooding Attacks**, Abraham Yaar, Adrian Perrig, Dawn Xiaodong Song, IEEE Symposium on Security and Privacy 2004
NOTE: You can find most of these papers at:
http://www.utdallas.edu/%7Eksarac/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