Syllabus for Part II of the Ph.D. Qualifying Exam in Computer Science Revised (9/21/18)

Part II of the Ph.D. qualifying exam is also for four hours. Students will be tested on the two specialty areas they chose when they registered for the exam. Each area will have four questions; a student must answer any three of them. The exam is closed-book and closed-notes. The use of electronic devices is not allowed.

This document describes the topics from which the questions for each specialty area will be chosen. Each area is accompanied by (i) suggested texts and (ii) Computer Science courses that cover the area in sufficient depth.

Artificial Intelligence

(Solve 3 questions.)

CS 422 (Introduction to Artificial Intelligence) or **CS 710 (Artificial Intelligence)** (2 questions)
Basics of state-space search (terminology (initial state, goal, operators, successor function, nodes (root, parent, child, sibling, leaf, terminal, ancestors, descendants), path cost, search cost, branching factor, node expansion, search tree, termination condition), evaluation criteria for searches (completeness, optimality, time complexity, and space complexity), **blind searches** (depth-first search, breadth-first search, uniform-cost search, depth-limited search, iterative-deepening search, and bidirectional search (algorithms, completeness, optimality, time complexity, and space complexity of these searches)), **informed searches** (pure heuristic search, A^*, hill climbing, simulated annealing (algorithms, completeness, optimality, time complexity, and space complexity of these searches), admissibility of heuristics, and local minima), **adversarial search** (Minmax algorithm, Shannons modifications to Minmax algorithm, evaluation functions, and alpha-beta pruning), **knowledge representation** (syntax, semantics, and rules of inference of propositional logic and first-order logic, encoding in these logics, limitations of these logics, different forms of sentences (Horn form, disjunctive normal form (DNF), and conjunctive normal form (CNF)), inference procedures including refutation, data-driven reasoning, and goal-directed reasoning), **state-space planning** (assumptions in classical planning, STRIPS, operators and actions, algorithms and properties of forward state-space planning and backward state-space planning, plan verification), and **natural language processing** (phrase structure grammars, network grammars, and chart parsing)

Suggested Reference:
Stuart Russell and Peter Norvig, Artificial Intelligence: A Modern Approach, 3rd ed., Prentice Hall, 2010. Chapters 3, 4.1, 5:1-4, 7, 8, 9, 10:1-2, 22, 23:13

CS 720 (Computational Models of Decision Making) (1 question)
Types of probability distributions for discrete variables (prior, conditional, full, partial, and joint), **Laws of probability, Bayesian networks with discrete variables** (writing expressions for queries, finding probabilities efficiently, and simplifying queries), **decision theory** (constructing a decision tree with chance nodes and decision nodes, solving this decision tree to find optimal policy or decision, constructing a payoff matrix, finding the best decision using expected utilities, sensitivity analysis, expected value of perfect information (EVPI), and expected value of sample information (EVSI)), **game theory** (two-person constant-sum games, converting a two-person constant-sum game into a zero-sum game, pure strategy, mixed strategy, equilibrium, stable and unstable games, minimax and maximin values, dominant and dominated strategies, and simplifying a payoff matrix by removing dominated strategies)

Suggested References:
(i) Stuart Russell and Peter Norvig, Artificial Intelligence: A Modern Approach, 3rd ed., Prentice Hall, 2010. Chapters 13, 14, and 16.
(ii) R. Bronson and G. Naadimuthu, Schaum's Outline of Operations Research, 1997. Chapters 17 and 18.

CS 711 (Introduction to Machine Learning) (1 question)

1. Review of probabilities and random variables (chapters 1-2 of [1], chapter 2 of [4])
2. Decision trees (chapter 4 rom [5])
3. Similarity-based learning (nearest-neighbor method) (chapter 5 from [5])
4. Linear and logistic regression (chapters 7 and 8 of [4])

5. Support vector machines (chapter 14 of [4])
6. Ensemble learning (chapter 16 of [4])
7. Neural networks and deep learning (chapter 5 of [3], chapter 28 of [4], and chapters 6-12 of [2])
8. Clustering, mixture models and the EM algorithm (chapters 11 and 25 of [4])

Suggested References:

[1] H. Stark and J. W. Woods, Probability, Statistics, and Random Processes, 4th Edition, Pearson, 2012.
[2] I. Goodfellow et al. Deep Learning, MIT Press, 2016.
[3] C. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.
[4] Kevin P. Murphy, Machine Learning, MIT Press, 2012
[5] John D. Kelleher, Brian Mac Namee, and Aoife D'Arcy, Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies, The MIT Press, 2015.

Computer Architecture and Performance Evaluation

Computer Architecture

Topics: central processing unit design, instruction set design, control unit design, arithmetic and logical (ALU) design, pipelining and vector processing, input/output organization, memory organization, parallel computer architecture, multi-core computer architecture.

Suggested Texts:
(i) M. Morris Mano, Computer Systems Architecture, 3rd ed., Prentice Hall, 1992. Chapters 5–13.
(ii) William Stallings, Computer Organization and Architecture, 9th ed., Prentice Hall, 2012. Chapters 17–18.

Course: CompSci 458: Computer Architecture

Computer Systems Performance Evaluation

Topics: discrete and continuous random variables, expectation, conditional expectation, stochastic processes, discrete and continuous time Markov chains, networks of queues.

Suggested Text: Kishor Trivedi, Probability and Statistics with Reliability, Queuing, and Computer Science Applications, 2nd ed., John Wiley and Sons, 2001. Chapters 1–9.

Course: CompSci 760: Computer Systems Performance Evaluation

Computer Graphics and Visualization

Topics: graphics primitives and attributes, line-drawing algorithms, 2D and 3D geometric transformations, 2D and 3D viewing, 3D object representations (Hermite splines, Bezier splines, B-splines), visibility detection, illumination models and polygon rendering, texture mapping, color models, ray-tracing, and basic computer animation.

Suggested Text: Donald Hearn and M. Pauline Baker, Computer Graphics with OpenGL, 4th ed., Prentice Hall, 2010. Chapters 1–10, 12, 13, 14, 16, 17, 18, 19, 21.

Courses: CompSci459: Fundamentals of Computer Graphics and CompSci:718: Advanced Computer Graphics: Modeling and Animation

Networks

The Computer Networks portion of the qualifying exam covers the fundamental concepts in wired and wireless networking, hardware technology, and software protocols.

Topics: physical layer, data communications, wired and wireless transmission media , data link layer, wired and wireless local area networks, wide area networks, network layer, routing protocols, mobile IP, transport layer , TCP/IP protocol suite, bluetooth, ad hoc and wireless sensor networks, multicast routing , congestion control and traffic management, and quality of service (QOS).

Suggested Text: Williams Stallings, Data Computer Communications, 9th ed., Prentice Hall, 2011. Chapters: 1–22.

Courses: CompSci 520: Computer Networks and CompSci 730: Advanced Computer Networks

Computer Security

Topics: components of computer security and classification of security threats, symmetric key and public key cryptosystems, cryptographic hash functions, digital signatures, key managements, network security protocols, entity authentication, system security and computer viruses.

Suggested Texts:
(i) Michael Goodrich and Roberto Tamassia, Introduction to Computer Security, Addison-Wesley, 2011. Chapters 1, 3–8.
(ii) William Stallings, Cryptography and Network Security: Principles and Practice, Pearson, 2017. Chapters 1–20.

Courses: CompSci 469: Intro to Computer Security and CompSci 759: Data Security

Human Computer Interaction and User Interfaces

Natural Language Processing

Topics: words, syntax: grammars and parsing (syntactic and statistical), semantics, semantic analysis, and word sense discrimination, discourse, applications (dialog, information extraction, factoid question answering).

Suggested Texts:
(i) Dan Jurafsky and James Martin, Speech and Language Processing, 2nd ed., Prentice Hall, 2009. Chapters 3.1, 3.2, 3.8, 3.9, 4.2, 5.1, 5.2, 12.1-12-4, 13.1-13.4, 14.1-14.3, 14.6.1, 14.7, 17.1-17.4, 18.1, 18.2, 18.5, 19.1, 20.1, 21.1, 21.2, 22.1, 22.2, 23.2, 24.1, 24.2.
(ii) Steven Bird, Ewan Klein, and Edward Loper, Natural Language Processing with Python — Analyzing Text with the Natural Language Toolkit, O'Reilly Media, 2009. Also available online at `http://www.nltk.org/book`. Preface, Chapters 3.7, 3.8, 5, 7.2, 7.3, 8.1-8.4, 10.1-10.3, 10.5, 7.1.

Course: CompSci 423: Introduction to Natural Language Processing, CompSci 723: Natural Language Processing, CompSci 444: Intro to Text Retrieval, or CompSci 744: Text Retrieval

Human-Computer Interaction

Topics: requirements gathering and specification, design concepts and components, designing for GUIs and the web, evaluation planning and analysis, advocating usability.

Suggested text: Debbie Stone, Caroline Jarrett, Mark Woodroffe and Shailey Minocha, User Interface Design and Evaluation, Morgan Kauffman, 2005. All chapters.

Course: CompSci 743: Intelligent User Interfaces or CompSci 747: Human-Computer Interaction

Programming Languages and Compilers

The programming languages and compilers portion of the qualifying examination covers the topics of programming languages concepts, compilers and type theory.

Programming Language Concepts

Topics: Syntax, BNF grammar, language systems, types, scopes, memory location for variables, polymorphism, memory management, parameter passing, formal semantics, functional programming (ML language, pattern matching, higher order functions, data types), object-oriented programming (classes, abstract classes and interfaces, iterators, subtype polymorphism), logic programming (Prolog language, unification, resolution, optimization).

Suggested Text: Adam Webber. Modern Programming Languages: A Practical Introduction, Franklin, Beedle and Associates, 2003. All chapters.

Course: CompSci 431: Programming Languages Concepts

Compilers

Topics: compiler phases (scanning, parsing, semantic analysis, machine-independent optimization, machine dependent optimization, code generation), regular expressions, parsing tables, symbol tables, type rules, attribute grammars, pipeline issues, flow analysis, separate compilation, linking.

Suggested Text: Michael Scott. Programming Languages Pragmatics, 3rd edition, Morgan Kaufmann, 2009. Chapters 1–9, 14, 16.

Course: CompSci 654: Introduction to Compilers, or CompSci 754: Compiler Construction

Type Systems

Topics: operational semantics, type rules, lambda calculus, proofs (progress, preservation, inversion, canonical forms), type concepts (products, sums, records, mutable state, subtyping, recursive types, universals, existentials, bounded quantification, higher-order polymorphism).

Suggested Text: Benjamin Pierce, Types and Programming Languages, MIT Press, 2002. All chapters excepting 4, 7, 10, 12, 17, 25.

Course: CompSci 732: Type Systems for Programming Languages

Theory

The theory portion of the exams covers the topics of automata and formal languages, and algorithm design and analysis.

Automata and Formal Languages

Topics: finite automata, pushdown automata, Turing machines and variants, regular languages, context-free languages, recursive languages, recursively enumerable languages, regular expressions, various classes of grammars and normal forms, reducibility, decidability.

Suggested Texts:
(i) Michael Sipser, Introduction to the Theory of Computation, 3rd ed., Thomson Course Technology, 2013. Chapters 1-5.
(ii) Peter Linz, An Introduction to Formal Languages and Automata, 5th ed., Jones and Bartlett, 2011. Chapters 1-12.

Course: CompSci 417: Introduction to the Theory of Computation

Algorithm Design and Analysis

Topics: asymptotic notation, solving recurrence relations, stacks, queues, vectors, lists, trees, priority queues and heaps, hashing, binary search trees (including AVL trees and red black trees), sorting and selection, structures/algorithms for disjoint sets, algorithm design techniques (the greedy method, divide and conquer, recursion, dynamic programming), graph algorithms (including minimum spanning tree algorithms and shortest path algorithms).

Suggested Texts:
(i) Thomas Cormen, Charles Leiserson, Ronald Rivest and Clifford Stein, Introduction to Algorithms, 3rd ed., The MIT Press, 2009. Chapters 3, 4, 6, 7, 10–13, 15, 21–25.
(ii) Michael Goodrich and Roberto Tamassia, Algorithm Design, John Wiley and Sons Inc., 2002. Chapters 1.1–1.4, 2 to 7.

Course: CompSci 535: Algorithm Design and Analysis

Analysis of Algorithms

Topics: selection, amortized analysis, algorithm design techniques (the greedy method, divide and conquer, recursion, dynamic programming), graph algorithms (including minimum spanning tree algorithms, shortest path algorithms and network flows), randomized algorithms (including quicksort, the closest pair of points problem, the hiring problem, the Rabin-Karp string matching algorithm, primality testing), NP-completeness, approximation algorithms (including the vertex cover problem, the metric TSP, set covering).

Suggested Texts:
(i) Thomas Cormen, Charles Leiserson, Ronald Rivest and Clifford Stein, Introduction to Algorithms, 3rd ed., The MIT Press, 2009. Chapters 3–5, 9, 15–17, 26, 34, 35.
(ii) Jon Kleinberg and Éva Tardos, Algorithm Design, Addison-Wesley, 2005. Chapters 3–8, 11, 13.
(iii) Sanjoy Dasgupta, Christos Papadimitriou and Umesh Vazirani, Algorithms, McGraw Hill, 2008. Chapters 2–8, 9.2.

Course: CompSci 704: Analysis of Algorithms