DEPTH (QUALIFIER) EXAMINATION IN THEORY OF COMPUTATION AND ALGORITHMS

Spring, 2007

Automata, Languages and Complexity:

1. Regular languages
   (a) finite state automata (deterministic, nondeterministic)
   (b) minimization
   (c) regular (right-linear) grammars
   (d) the pumping lemma
   (e) regular expressions
   (f) analysis: automaton → regular expression
   (g) synthesis: expression → automaton
   (h) pattern matching automata
   (i) closure properties
2. Context-free languages
   (a) context-free grammars
   (b) normal forms (Chomsky, Greibach)
   (c) CYK algorithm
   (d) the pumping lemma
   (e) existence of ambiguous languages
   (f) pushdown automata
   (g) closure properties
3. Recursive and r.e. languages
   (a) Turing machines
   (b) non-restricted grammars
   (c) recursive and partial recursive functions
   (d) Church's Thesis
   (e) universal Turing machine
   (f) diagonalization
   (g) (Turing and many-one) reductions
   (h) undecidability of the halting problem
   (i) Rice's theorem for recursive sets

(j) Post's correspondence problem

(k) Hilbert's 10th problem

(l) undecidable problems for context-free languages

(m) other undecidable problems

(n) closure properties

4. Complexity theory

   (a) time and space bounded computation

   (b) determinism vs nondeterminism

   (c) complexity classes

   (d) space and time hierarchies

   (e) Savitch's theorem

   (f) NP-completeness and NP-hardness

   (g) methods for proving and dealing with NP-hard problems

   (h) problems complete in NL, P, PSPACE

Algorithms and Data Structures:

1. Basic data structures: lists, queues and stacks
2. Trees and tree traversal methods
3. Sorting and selection

   (a) internal sorting algorithms: insertion sort, bubblesort, mergesort, heapsort, quicksort

   (b) linear time algorithm for selection

4. Binary search trees and their applications

   (a) AVL trees

   (b) red-black trees

   (c) B-trees

   (d) splay trees

5. Priority queues, heaps and applications
6. String algorithms

   (a) Knuth-Morris-Pratt string searching

   (b) Boyer-Moore string searching

   (c) Huffman codes

7. Union-Find
8. Graph algorithms
    (a) depth-first-search, breadth-first-search, connectivity
    (b) minimum spanning trees
    (c) shortest path algorithms (single source, all-pairs)
    (d) network flow
    (e) bipartite graph matching
9. Number-theoretic problems and cryptography
    (a) exponentiation by doubling
    (b) Euclid's algorithm
    (c) primality testing
    (d) RSA
10. Algorithm design and analysis techniques and their applications
    (a) asymptotic analysis; recurrence relations.
    (b) divide and Conquer (including FFT and Strassen's algorithm)
    (c) dynamic programming
    (d) greedy algorithms
    (e) branch and bound
    (f) basic approximation algorithms
    (g) basic probabilistic algorithms
    (h) basic parallel algorithms
11. Lower bound techniques
    (a) decision trees for sorting and selection
    (b) adversary (*i.e.* oracle) arguments

This list is intended to be indicative but not exhasutive. The exam may include some questions on material not covered in either the graduate or the undergraduate courses. However, the references below are quite complete; it is very unlikely that questions outside their scope would occur.

The kinds of questions may include problems, proofs, statements of fundamental definitions and results, multiple choice, and true/false. Negative credit will be awarded for seriously misinformed answers.

Since it is only possible to judge someone's answers and not their mind, you are advised to practise writing down answers to questions in precise and intelligible English. Many points are lost through improperly worded answers by students who probably understand the concepts and methods necessary for a correct answer.

Many students have found it helpful to discuss the material to be covered with fellow graduate students. This has three benefits: it reinforces the material in the minds of both students; it offers an opportunity to correct misimpressions concerning the material (but also to propogate them); it gives the students practice in phrasing answers. It is advisable to practise writing answers out as well as discussing them.

Copies of previous qualifier exams (which were called comprehensive exams before) are in *marek/PUBLIC/THEORY_*
The committee for the depth examination consists of Profs. Chrobak, Jiang, Lonardi, and Young. If you have any questions regarding the test, please contact Prof. Jiang. A review session will be scheduled prior to the exam. If you are planning to attend it, it is recommended that you submit a list of questions to Prof. Jiang at least a week in advance. Anyone who plans to take the exam should notify Prof. Jiang and Terri by email before April 10, 2007.

References

[1] J. Hopcroft, J. Ullman, Introduction to Automata Theory, Languages and Computation, Addison Wesley, 1979.

[2] J. Hopcroft, R. Motwani, J. Ullman, Introduction to Automata Theory, Languages and Computation, 2nd edition, 2001, Addison-Wesley.

[3] M. Sipser, Introduction to the Theory of Computation, PWS Publishing Co, 1996.

[4] H. Lewis, C. Papadimitriou, Elements of the Theory of Computation, Prentice Hall, 1981.

[5] T.H. Cormen, C. Leiserson, R. Rivest, C. Stein, Introduction to Algorithms, McGraw-Hill, 2nd edition, 2001. A. Levitin. Introduction to the Design and Analysis of Algorithms, 2nd edition, 2007, Addison Wesley.

[6] A.V. Aho, J.E. Hopcroft, J.D. Ullman, Data Structures and Algorithms, Addison-Wesley, 1983.

[7] A.V. Aho, J.E. Hopcroft, J.D. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley, 1974.

[8] M.R. Garey,D.S. Johnson, Computers and Intractability - A Guide to the Theory of NP-Completeness, W.H.Freeman and Co, 1979.

[9] G. Brassard, P. Bratley, Fundamentals of Algorithms, Prentice Hall, 1996.

[10] C. Papadmimitriou, Computational Complexity, Addison-Wesley, 1994.

[11] D. Kozen, The Design and Analysis of Algorithms, Springer-Verlag, 1991.

[12] M. Minsky, Computation: Finite and Infinite Machines, McGraw Hill, 1967.