CSC 342 - ALGORITHM ANALYSIS

CREDIT HOURS: 3
PREREQUISITES: CSC 214 and 241
GRADE REMINDER: Must have a grade of C or better in each prerequisite course.

CATALOG DESCRIPTION

Study of algorithm design, analysis tools and techniques for selected problems, including sorting, searching, graphs, branch and bound strategies, dynamic programming, algebraic methods, string matching, and sets. An introduction to order notation, timing routines and complexity classes.

PURPOSE OF COURSE

The purpose of this course is to provide the student with tools and techniques for analyzing problem solutions. Complexity theory and computability issues are introduced. Evaluation of algorithms used in solving representative problems will be emphasized.

EDUCATIONAL OBJECTIVES

This course will provide students an opportunity to do the following:

1. To develop the concept of an algorithm, and thereby distinguish between solvable and unsolvable problems.

2. To present various complexity-levels of algorithms, and illustrate the concept with examples of algorithms that run in polynomial time as well as some that require exponential time.

3. To apply formal analysis techniques, based on algorithm time and space requirements, to algorithms involving iteration and recursion.

4. To develop the use of mathematical techniques, such as recurrence relations, as tools for analyzing the complexity of algorithms.

5. To study, implement, and analyze the performance of algorithms for sorting, generalized searching, string matching, pattern matching, and data compression.

6. To develop and implement branch-and-bound algorithms for solving selected NP-complete problems, and present efficient heuristic methods for finding sub-optimal but practical solutions to such problems.

7. To discuss emerging trends in algorithm developments, including parallel and distributed processing.

CONTENT

<table>
<thead>
<tr>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Introduction and Math Preliminaries</strong></td>
</tr>
<tr>
<td>Notation, recursion, recurrence, induction</td>
</tr>
<tr>
<td>Order notation</td>
</tr>
<tr>
<td>Analysis tools and timing routines</td>
</tr>
<tr>
<td>Overview of complexity classes (P, NP, exponential)</td>
</tr>
</tbody>
</table>

| **Sorting and Searching Algorithms** | 5 |
| Selection, insertion, exchange, special, binary, tree, hashing |

| **String Algorithms** | 3 |
| Matching (Knuth-Morris-Pratt, Boyer-Moore), parsing, compression |
Graph Algorithms .......................................................... 9
  Representation, connectivity, reachability, traversal, shortest path, minimum spanning tree,
  transitive closure, topological sort, Steiner trees, networks, depth first search, breadth
  first search.

Set Algorithms ...................................................................... 3
  Union-find, dictionary, Boolean matrices

Algebraic Methods .................................................................. 3
  Polynomials, matrix operations, random numbers

Complexity Classes ................................................................. 4
  Definition, classes, examples

Branch and Bound Algorithms .................................................. 3
  Backtracking, greedy methods, traveling salesman problem

Dynamic Programming ............................................................ 3
  Optimal search trees, all pairs shortest path

Advanced Topics ...................................................................... 4
  Computation models - computability, decidability, finite state machines, grammars,
  pushdown automata
  Iterative Refinement
  Parallel and Distributed algorithms
  Geometric algorithms
  Heuristics (including genetic and neural techniques)
  Probabilistic and approximation algorithms

Exams ................................................................................... 3

TOTAL 45

REFERENCES

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


Hill, 2009.


