ARIZONA STATE UNIVERSITY

CSE 310 SLN 70860 — **Data Structures and Algorithms** — Fall 2015

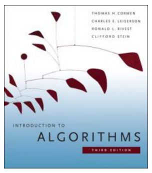
Instructor:	Dr. Hanghang Tong (e-mail: `htong6@asu.edu`; office: BYENG 416)
Lectures:	Mondays/Wednesday 6:00-7:15pm, in COOR L1-74
Office Hours:	Mondays/Wednesday 10:00-11:00am, and by appointment only
TA:	Joydeep Banerjee, office hours: Tuesday/Friday 1:00-2:00pm, Center Point: 114

Course Description

CSE 310 studies how data structures impact the computational complexity (time and space) of algorithms. Data structures covered include stacks, queues, heaps, linked lists, search trees and their balanced variants, hash tables, disjoint sets, and graphs. Algorithmic techniques covered include divide-and-conquer, dynamic programming, and greedy. While most algorithms are analyzed in the worst or average case, also of interest are lower bounds, and amortized and probabilistic analyses. An introduction to concepts in parallel and distributed algorithms is planned.

Prerequisites: For CS/CSE students, grades of at least C in CSE 220 or CSE 240 and at least D in MAT 243. For CMS students, grades of at least C in CSE 210 and at least D in MAT 243 or MAT 300. Programming experience in `C` or `C++` is expected.

Required Textbook



Introduction to Algorithms, Third Edition, by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein. © 2009 by Massachusetts Institute of Technology. ISBN 978-0-262-03384-8.

Evaluation Procedure

Assignments	10%	4 of equal weight
Projects	15%	3 of equal weight, all in `C/C++` only on `general.asu.edu` (Linux)
Midterm Exams:	30%	2 of equal weight, in class (closed book)
Final Exam:	45%	Monday, December 7, 2015, 9:50-11:40am (closed book, comprehensive)
	100%	

Assignments	Out	Due	Projects	Out	Milestone	Due	Midterms	Scheduled
A1	08/24	09/11	P1	08/24	09/09	09/23	M1	09/30
A2	09/16	10/09	P2	09/23	10/16	10/28	M2	11/09
A3	10/14	11/06	P3	10/28	11/20	12/04		
A4	11/06	11/25						

For consistency, all sections of CSE 310 have the same assignments and projects on the same schedule. Midterm and final exams, however, will be set by each instructor.

Course Schedule (Subject to Change)

Foundations: (approximately 2 weeks)

- The Role of Algorithms in Computing (Chapter 1): What is an algorithm?
- Getting Started (Chapter 2): Insertion sort. Analyzing and designing algorithms. Mergesort.
- Growth of Functions (Chapter 3): Asymptotic notation. Standard notation and common functions.
  - Introduction to parallel time and space complexity.
- Divide-and Conquer (Chapter 4): Selected problems solved using divide-and-conquer. The substitution, recursion tree, and master method for solving recurrences.
  - Parallel aspects of divide-and-conquer (e.g., mergesort, numerical integration).

Sorting and Order Statistics: (approximately 2 weeks)

- Heapsort (Chapter 6): Heaps and maintaining the heap property. The heapsort algorithm.
- Quicksort (Chapter 7): The quicksort algorithm and its analysis. Randomized quicksort.
- Medians and Order Statistics (Chapter 9): Minimum and maximum. The selection problem.
  - Parallel algorithms for finding order statistics, notably min and max.

Data Structures: (approximately 2 weeks)

- Elementary Data Structures (Chapter 10): Stacks and queues. Pointers and objects. Implementing lists and rooted trees.
- Hash Tables (Chapter 11): Direct-address tables. Hash tables and functions. Open addressing.
- Binary Search Trees (Chapter 12): What is a binary search tree? Queries, insertion, and deletion.
  - Parallel search and possibly parallel updates in trees.
- Red-Black Trees (Chapter 13): Properties of red-black trees. Rotation, insertion, and deletion.

Advanced Design and Analysis Techniques: (approximately 2 weeks)

- Dynamic Programming (Chapter 15): Selected problems solved using dynamic programming.
- Greedy Algorithms (Chapter 16): Selected problems solved using the greedy approach.
- Amortized Analysis (Chapter 17 and 21): Disjoint-set operations and their amortized analysis.

Graph Algorithms: (approximately 3 weeks)

- Elementary Graph Algorithms (Chapter 22): Representations of graphs. BFS and DFS.
  - How to carry out BFS- and DFS-like parallel search in a graph or solution space.
- Minimum Spanning Trees (Chapter 23): The algorithms of Kruskal and Prim.
- Single-Source Shortest Paths (Chapter 24): The Bellman-Ford algorithm and Dijkstra's algorithm.

Use of Blackboard

Blackboard will be used in CSE 310. It is accessible from a browser at `http://my.asu.edu`. It contains the syllabus and lecture notes. All homework assignments and programming projects will be posted on **Blackboard**. All assignments and projects must be submitted electronically on **Blackboard**. Use the discussion group to ask questions about anything related to CSE 310; you are expected to check it often and to participate.

Class Policies

Assignments and projects are due electronically at *midnight* on their due date. Late assignments and projects will **not** accepted, except on a documented emergency basis. For consideration of an extension you must contact the instructor *prior* to the due date. Similarly for rescheduling a midterm. The final exam date is scheduled by ASU and cannot be changed.

Whenever a new grade for work is available it will be posted on **Blackboard**. If you wish to appeal the grade, you must do so *in writing* within one week of the grade availability. Your right to appeal is waived one week after the grade is posted. It is imperative that you make a legitimate attempt to do all assigned work. Any scaling of grades at the end of the course takes into account the effort invested.

ASU Code of Conduct and Academic Integrity Policy

Plagiarism or any form of cheating in assignments, projects, or exams is subject to serious academic penalty; this may range from a grade of zero for the work to failure of the course. To understand your responsibilities as a student at ASU read:

- Student Code of Conduct and Student Disciplinary Procedures: `http://students.asu.edu/srr/code`
- Student Academic Integrity Page: `http://provost.asu.edu/academicintegrity`
- See also the content under the "Collaboration: What is and is not permitted" folder under the *Syllabus & Course Information* tab on Blackboard.

Because all sections of CSE 310 have common assignments and projects, all submissions from all sections will be cross-checked with each other.

CSE 310 Course Learning Outcomes

Students who complete CSE 310 will be able to:

1. Define data structures such as heaps, balanced trees, and hash tables.
2. Explain how to use a specific data structure in modelling a given problem.
3. Identify, construct, and clearly define a data structure that is useful for modelling a given problem.
4. State some fundamental algorithms such as merge sort, topological sort, Prim's and Kruskal's algorithm, and algorithmic techniques such as dynamic programming and greedy algorithms.
5. Use a specific algorithmic technique in solving a given problem.
6. Design an algorithm to solve a given problem.
7. Define the notions of worst-, best-, and average-case running times of algorithms.
8. Analyze and compare different asymptotic running times of algorithms.
9. Analyze a given algorithm and determine its asymptotic running time.
10. Combine fundamental data structures and algorithmic techniques in building a complete algorithmic solution to a given problem.
11. Create several algorithmic solutions to a given problem and choose the best one among them according to given requirements on time and space complexity.

Additional outcomes related to parallel and distributed computing:

1. Be exposed to models and the intrinsic degree of parallelism of some elementary key algorithms.
2. Follow arguments for parallel time and space complexity.
3. Observe how the structure that enables divide-and-conquer (sequential) algorithms exposes opportunities for parallel computation.
4. Observe algorithms for finding order statistics. Understand that selection can always be accomplished by sorting but that direct algorithms may be simpler.
5. Know how to carry out BFS-like parallel search in a graph or solution space.