0. Class Meeting Times/Places

1. Lecture: Monday/Wednesday 4:00 – 5:15 pm; Agricultural Hall 202

1. General Information

Instructor: H. K. Dai
Teaching Assistant: A. Vuduthala

Office Location: Mathematics, Statistics, and Computer Science Building
Room 209

Office Hours: Tuesday/Thursday 4:30 – 5:30 pm
Office Phone: 744-7207
email Address: dai@cs.okstate.edu

2. Course Description in Current University Catalog

CS 3353: Data Structures and Algorithm Analysis I. Prerequisites: CS 2133 (Computer Science II) and CS 3653 (Discrete Mathematics for Computer Science). Storage, structures, data and information structures, list processing, trees and tree processing, graphs and graph processing, searching, and sorting.

3. Course Goals

Study of analysis and implementation of data structures and algorithms. Basic techniques for asymptotic analysis of algorithm-time bounds. Fundamental and advanced data structures. Basic algorithm-design techniques such as divide-and-conquer, dynamic programming, and greedy techniques for optimization. Applications of these design and analysis techniques to fundamental problems such as sorting, searching, and graph algorithms.

4. Course Materials/References


5. Course Website and Lecture Notes

Our course Website is maintained on the campus-wide online learning system Canvas, which can be accessed through “https://canvas.okstate.edu”.

Since the information in our class pages will be constantly updated, please check in Canvas regularly (Announcements, Assignments, Modules, etc.).

General notes:

1. All class materials (announcements, lecture notes, assignments, etc.) will be disseminated on Canvas.

6. Homework and Examinations

There will be about six homework assignments, one test, and one final examination.

7. Course Grade

The course grade is based on the homework (40%), test (25%), and final examination (35%). The passing letter-grade is determined by the following partition of the course grades:

D : [50, 60); C : [60, 70); B : [70, 85); and A : [85, 100]

8. Miscellaneous

1. Lectures: Lectures are not mandatory, but historically, students with active attendance/coursework have done significantly better on examinations than their less frequently attending classmates.
2. **Homework**: Problem sets form an important part of the learning in the course, and thus, you are required to do them in order to pass.

3. **Collaboration and Sharing**: You are encouraged to discuss approaches with other students on solutions of assigned coursework, but you must write up solutions on your own independently and acknowledge your source in the write-up for each problem. If you obtain a solution with help (e.g., through library or publicly available work, or academic work by other students — whether in this or previous semesters), acknowledge your source, and write up the solution on your own.

   **Notes**: Read relevant documents/guidelines about academic integrity at Oklahoma State University in Academic Integrity Resources at the following URL:
   https://academicaffairs.okstate.edu/academic-integrity/index.html

9. **Student Disability Services**

   Student Disability Services and other Student Services are committed to providing support services to students with physical and learning disabilities. Please advise the instructor of desired academic accommodations, and notify Student Disability Services.

10. **Academic Dishonesty or Misconduct**

   Refer to the section in “University Academic Regulations” in current “University Catalog” (http://registrar.okstate.edu/)

11. **Adding/Dropping/Withdrawing, Important Dates, and Syllabus Attachment**

   1. **Test and Final Examination**: Tentative date for the test is October 4 (Monday), 2021.

      Adopting “Fall 2021 Final Exam Schedule”, the firm time/date for final examination is 6:00 – 7:50 pm, December 6 (Monday), 2021 in regular class meeting place.

      Refer to the section in “Fall 2021 Final Exams”:
      https://registrar.okstate.edu/class_schedule_short_courses/exams.html

   2. **Adding/Dropping/Withdrawing and Important Dates**: Refer to the section in “Academic Calendar”:
      http://registrar.okstate.edu/

   3. **Syllabus Attachment**: Refer to:
      https://academicaffairs.okstate.edu/student-support/index.html
1. ABET (Accreditation Board for Engineering and Technology, Inc.): Student Outcomes

1. Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions.

2. Design, implement, and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program’s discipline.

6. Apply computer science theory and software development fundamentals to produce computing-based solutions. [CS]

2. ACM (Association for Computing Machinery): The Body of Knowledge

<table>
<thead>
<tr>
<th>Knowledge Area</th>
<th>Total Hours of Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software Development Fundamentals (SDF)</td>
<td>2</td>
</tr>
<tr>
<td>Programming Languages (PL)</td>
<td>1</td>
</tr>
<tr>
<td>Algorithms and Complexity (AL)</td>
<td>34</td>
</tr>
</tbody>
</table>
3. Body of Knowledge Coverage

<table>
<thead>
<tr>
<th>Knowledge Area</th>
<th>Knowledge Unit</th>
<th>Topics Covered</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDF</td>
<td>algorithms and design</td>
<td>concept and properties of algorithms, role of algorithms, problem-solving strategies, separation of behavior and implementation</td>
<td>1</td>
</tr>
<tr>
<td>SDF</td>
<td>fundamental data structures</td>
<td>stacks, queues, linked structures</td>
<td>1</td>
</tr>
<tr>
<td>PL</td>
<td>objected-oriented programming</td>
<td>objected-oriented design, encapsulation, iterators</td>
<td>1</td>
</tr>
<tr>
<td>AL</td>
<td>basic analysis</td>
<td>all core-tier 1: best-, expected-, and worst-case behaviors of an algorithm, asymptotic notions/notations (big-O), complexity classes such as constant/logarithmic/polynomial/exponential, empirical measurements of performance, time/space trade-offs in algorithms all core-tier 2: big-O/Omega/Theta and little-O/Omega notions/notations and asymptotic analysis, recurrence relations and some version of Master Theorem, analysis of iterative and recursive algorithms</td>
<td>6</td>
</tr>
<tr>
<td>AL</td>
<td>algorithmic strategies</td>
<td>core-tier 1: brute-force, greedy approach, recursive divide-and-conquer, dynamic programming core-tier 2: heuristics</td>
<td>6</td>
</tr>
<tr>
<td>AL</td>
<td>fundamental data structures and algorithms</td>
<td>all core-tier 1: simple numerical algorithms, binary search, sorting algorithms (selection, insertion, quicksort, heapsort, mergesort) and their worst- or average-case analyses, hashing, binary search trees and their common operations, graph-representations and simple graph-traversals core-tier 2: heaps, graph-algorithms: shortest-path algorithms (Dijkstra’s and Floyd’s algorithms), minimum spanning tree (Prim’s and Kruskal’s algorithms)</td>
<td>12</td>
</tr>
<tr>
<td>AL</td>
<td>basic automata computability and complexity</td>
<td>core-tier 2: introduction to the $P$, $NP$, and $NP$-complete classes</td>
<td>1.5</td>
</tr>
<tr>
<td>AL</td>
<td>advanced computational complexity</td>
<td>some classic $NP$-complete problems, reductions</td>
<td>1.5</td>
</tr>
<tr>
<td>AL</td>
<td>advanced automata theory and computability</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>AL</td>
<td>advanced data structures algorithms and analysis</td>
<td>balanced trees, B-trees, disjoint-sets, graph-algorithms: topological sort and strongly connected components</td>
<td>7</td>
</tr>
</tbody>
</table>
4. Course Outline (Tentative)

1. Mathematical Preliminaries and Introductory Material
   Source: Lecture Notes, and [CLRS09] Chapters 1, 2, 3, and 4
   1.1 Asymptotic Notations
   1.2 Recurrences

2. Sorting
   Source: Lecture Notes, and [CLRS09] Chapters 6, 7, and 8
   2.1 Heapsort, Mergesort, and Quicksort
   2.2 Lower Bounds for Sorting

3. Elementary Data Structures: Lists, Stacks, and Queues
   Source: Lecture Notes, and [CLRS09] Chapter 10

4. Hashing
   Source: Lecture Notes, and [CLRS09] Chapter 11
   4.1 Hash Functions
   4.2 Chaining
   4.3 Open Addressing

5. Trees, Tree Traversals, Binary Trees, and Search Trees
   Source: Lecture Notes, and [CLRS09] Chapter 12
   5.1 Trees and Their Implementations
   5.2 Binary Trees and Their Implementations
   5.3 Tree Traversals
   5.4 Binary Search Trees

6. Priority Queues and Their Implementations
   Source: Lecture Notes, and [CLRS09] Chapter 6

7. Advanced Design Techniques and Data Structures
   Source: Lecture Notes, and [CLRS09] Chapters 15, 16, and 18
   7.1 Dynamic Programming
   7.2 Greedy Algorithms
   7.3 B-Trees

8. Graph Algorithms
   Source: Lecture Notes, and [CLRS09] Chapters 22, 23, and 24
   8.1 Undirected and Directed Graphs, and Their Representations
   8.2 Some Fundamental Undirected and Directed Graph Algorithms
   8.3 Graph Traversals
   8.4 Minimum Spanning Trees
   8.5 Shortest-Path Problems