KAU CPCS 324 • 2020–21/Transitional

Algorithms and Data Structures (II)

Dr. Muhammad Al-Hashimi

Credits 3

Prerequisite CPCS 222, CPCS 223

Homepage www.hashimi.ws/cs324

Email cs324@hashimi.ws

Office Room 139 (B31) ✆ Ext. TBA

As specified by KAU syllabus, we study algorithms from a major application area and introduce advanced design techniques. A substantial graph implementation provides an opportunity to work with complex data structures and to develop advanced programming skills. We utilize group projects to promote teamwork and the use of relevant tools and practices. We discuss applications and ideas for capstone projects throughout the semester. Topics: dynamic programming, greedy and iterative improvement graph algorithms, introduction to limitations of algorithms, an approach to deal with those limitations, a brief intro to parallel algorithms.

Topics Current teaching schedule in homepage.

- ✎ **Graph Implementation** Directed and undirected weighted graph representations and input formats, traversals, connectivity and related properties.
- ✎ **Algorithms** Transitive closure, Warshall's algorithm, Floyd's all-pairs shortest-paths algorithm, Prim's and Kruskal's minimum spanning tree algorithms, Dijkstra's single-source shortest-paths algorithm, maximum flow problem, Ford-Fulkerson method, maximum-flow minimum-cut theorem, Edmonds-Karp shortest augmenting path algorithm, bipartite graphs and maximum bipartite matching, the stable marriage problem and more.
- ✎ **Data Structures** Priority queues, union-find, ~~hash tables~~, ~~B-trees~~, implementation of abstract data types.
- ✎ **Algorithm Design** ~~Trading space for time~~, dynamic programming, greedy techniques, iterative improvement, and (if time permits) an introduction to branch-and-bound.
- ✎ **Computational Complexity** Lower bounds, decision trees, polynomial efficiency, nondeterministc algorithms, decision problems, decidability and the halting problem, P, NP and NP-complete, impact of quantum computing.
- ✎ **Parallel Algorithms** TBA

Textbook (Chap. 7–12 Selective) Anany Levitin, *Introduction to The Design and Analysis of Algorithms*, Addison-Wesley (Pearson International Edition), 3rd edition, 2011. ISBN: 027376411X

Assessment Mostly programming projects (60% individual work, 40% group). Check homepage for curent evaluation criteria, deliverables, and deadlines. See note on flexi-articulation of course.

25%	Test 1 or Project 1
20%	Test 2 or [Group] Project 2
15%	Homework exercises
40%	Final**

**25% group project + 15% Project 1 Milestone 1

Learning Resources Check homepage for lecture schedule, assignments, supplemental material, and discussion group.

Learning Outcomes Detailed outcomes are articulated in course file. Broadly, students will:

1. Explore how complex data structures are implemented efficiently
2. Examine algorithms utilizing advanced techniques
3. Develop core technical code library for a capstone project
4. Consider limitations of algorithm power

Course Design Note Course components are designed to easily accommodate conventional assessments which can be selectively interchanged with programming projects. My version dispenses with exams and adopts a learn-by-doing approach instead.

References

1. Sedgwick, *Algorithms in C++ Part 5: Graph Algorithms*, Pearson Education, third edition, 2002 (Kindle Edition). ISBN: 0201361183
2. D. Lidar, *The Impact of Quantum Computing*, Fujitsu Labs Advanced Technology Symposium, 2017 (Talk 5/15/2018 on Youtube).
3. Steven Skiena, *The Algorithm Design Manual*, Springer, 2nd edition, 2008.

Rev 1.0 (8/30/2020)