

CMSC 451: Algorithms

Fall 2009

Course objectives: Study interesting computational problems and their algorithms, with a focus on the principles used to design those algorithms.

Instructor: Carl Kingsford, Office: AVW 3223. Email: carlk@cs.umd.edu. Office hours: Tuesdays 3:15-5:00. If you cannot attend office hours, email me about scheduling a different time.

Web page: <http://www.cs.umd.edu/class/fall2009/cmsc451/>

Class time: Mon/Wed 11:00-12:15 in CSIC 3120.

Final exam: Wednesday, Dec 16, from 8:00am to 10:00am in CSIC 3120.

Textbook: *Algorithm Design* by Jon Kleinberg and Éva Tardos. Addison Wesley, 2005. ISBN: 0-321-29535-8.

Course work: There will be ~10 homework sets drawn from the problems in the textbook. There may be additional homeworks or extra credit assignments. There will be 2 in-class exams, each non-cumulative, and a comprehensive final. Approximate grading weight: 10% for homeworks, 25% for each exam, and 40% for the final. The class will be graded on a curve.

Homework policies:

- Written problem sets are due at the start of class. **No late homework will be accepted** — turn in what you have completed. If you will miss class, turn in the homework early.
- Answers to homework problems should be written concisely and clearly. **Messy or poorly written homeworks will not be graded.** Typesetting homeworks with LaTeX is encouraged.
- Homework problems that ask for an algorithm should present: a clear English description or pseudocode, a proof that the algorithm is correct, and an analysis of the running time.
- Graded homeworks should be picked up in class; if you miss the class when the homework is returned, please pick it up during office hours.
- Regrade requests should be made in writing within 1 week of the homework being returned.
- The four problems for which you receive the lowest scores will be dropped.
- Exams and the final will be closed book, closed note. At least the first exam (and possibly the 2nd exam and the final) will be open homework: you may use **your own, graded homework solutions** during the test.
- You may discuss the problems with classmates. You must list the names of the class members with whom you worked at the top of your homework. **You must write up your own solution independently!**

Excused absences: Students claiming an excused absence must apply in writing and furnish documentary support (such as from a health care professional who treated the student) for any assertion that the absence qualifies as an excused absence. The support should explicitly indicate the dates or times the student was incapacitated due to illness. Self-documentation of illness is not sufficient to excuse the absence. Absences for religious observances must be submitted in writing to the instructor within two weeks of the start of the semester. The instructor is not under obligation to offer a substitute assignment or to give a student a make-up assessment unless the failure to perform was due to an excused absence. An excused absence for an individual typically does not translate into an extension for team deliverables on a project.

Academic accommodations: Any student eligible for and requesting reasonable academic accommodations due to a disability is requested to provide, to the instructor in office hours, a letter of accommodation from the Office of Disability Support Services (DSS) within the first two weeks of the semester.

Academic honesty: All class work should be done independently unless explicitly indicated on the assignment handout. You may *discuss* homework problems with classmates, but must write your solution by yourself. If you do discuss assignments with other classmates, you must supply their names at the top of your homework / source code. No excuses will be accepted for copying others work (from the current or past semesters), and violations will be dealt with harshly. (Getting a bad grade is much preferable to cheating.)

To quote the honor council: “The University of Maryland, College Park has a nationally recognized Code of Academic Integrity, administered by the Student Honor Council. This Code sets standards for academic integrity at Maryland for all undergraduate and graduate students. As a student you are responsible for upholding these standards for this course. It is very important for you to be aware of the consequences of cheating, fabrication, facilitation, and plagiarism. For more information on the Code of Academic Integrity or the Student Honor Council, please visit <http://www.shc.umd.edu>.”

To further exhibit your commitment to academic integrity, remember to sign the Honor Pledge on all examinations and assignments: “I pledge on my honor that I have not given or received any unauthorized assistance on this examination (assignment).”

Tentative Schedule & Reading

Introduction, Review, and Basics:

- 8/31: **Introduction & Stable Marriage Problem.** Chapter 1
- 9/2: **Mathematical background.** Chapter 2
- 9/9: **Graphs, DFS, BFS, Connectivity, DAGs, Topological sorting.** Chapter 3

Greedy Algorithms:

- 9/14: **Fractional knapsack & Interval scheduling.** 4.1, 4.2
- 9/16: **Interval scheduling, continued.** 4.2,4.3
- 9/21: **Dijkstra's shortest path.** 4.4
- 9/23: **Minimum spanning tree & clustering.** 4.5, 4.7

Divide and Conquer:

- 9/28: **MergeSort & Recurrence relations.** 5.1, 5.2
- 9/30: **Counting Inversions & Finding closest pair of points.** 5.3, 5.4

Dynamic Programming:

- 10/5: **Weighted interval scheduling.** 6.1, 6.2
- 10/7: **Subset Sum & Knapsack.** 6.4
- 10/12: **Sequence Alignment & RNA structure.** 6.5, 6.6
- 10/14: **Shortest paths in a graph.** 6.8
- 10/19: **MIDTERM #1.** Covers up through 10/12.

Network Flow:

- 10/21: **Linear programming.**
- 10/26: **Max flow & min cut.** 7.1–7.3
- 10/28: **Bipartite matching & disjoint paths.** 7.5, 7.6
- 11/2: **Circulations & survey design.** 7.7, 7.8
- 11/4: **Additional network flow applications.** some of 7.9–7.12

NP-hardness:

- 11/9: **Languages, the class NP, and reductions.** 8.1–8.3
- 11/11: **More example NP-completeness results.** 8.5–8.8
- 11/16: **Cook's Theorem, additional complexity classes.**

Approximation Algorithms & Heuristics:

- 11/18: **Load balancing & TSP.** 11.1
- 11/23: **Center selection problem.** 11.2
- 11/25: **Set Cover or Vertex Cover.** 11.3, 11.4
- 11/30: **Local search, simulated annealing.** 12.1, 12.2
- 12/2: **MIDTERM #2.** Covers from 10/14 to 11/25

Randomized Algorithms:

- 12/7: **Contention resolution.** 13.1
- 12/9: **Medians & Selection.** 13.5