CSCI 445

Analysis of Algorithms

Fall 2016 MFW 12:55–2:05 pm SCI 131

http://cs.wheaton.edu/~tvandrun/cs445

Thomas VanDrunen

☎630-752-5692 630-639-2255 ✉Thomas.VanDrunen@wheaton.edu

Office: SCI 163 Office hours: MTuWF 3:30–4:30pm; Th 9:30–10:30am and 1:00–3:00pm.

Contents

Catalog description. An introduction to algorithmic efficiency and to techniques for the design and analysis of efficient algorithms. General topics include review of asymptotics, algorithm design techniques (such as divide-and-conquer, dynamic programming, and greedy algorithms), graph algorithms, languages and automata, and NP-completeness.

Textbooks.

Cormen et al. *Introduction to Algorithms*, third edition, McGraw Hill / MIT Press, 2009. (See the course website for a link to solutions to some of the problems as found on the book's official website and a link to lectures by one of the book's authors from a class similar to ours.)

Lewis and Papadimitriou. *Elements of the Theory of Computation*, second edition, Prentice Hall, 1998. (The department owns copies for students to borrow.)

Objectives. The outcomes of this course fall under four headings: skills in the *analysis* of algorithms, *literacy* in well-known algorithms and data structures, skills in *devising and implementing* algorithms, and understanding of problems and results in *computational complexity*. In detail, and in relation to the computer science program outcomes, at the end of this course students should be able to

- analyze the time complexity of algorithms (CSCI PO D and E)
  - using worst-case and expected-case analysis
  - using recurrences and substitution
  - using the master method
  - using amortized analysis
- identify, articulate, and adapt algorithms and data structures (CSCI PO A and F)
  - from classical problems such as sorting and searching
  - from a variety of topics chosen by the instructor after considering students' interests
- devise and implement advanced algorithms (CSCI PO A and F)

  The other three goals are linked to the three units of the course, but this goal—improving students' algorithm-writing ability—will be pursued and assessed throughout the semester.
- identify and categorize intractable problems (CSCI PO C, D, and J)
  - by proving problems to be undecidable
  - by proving problems to be $\mathcal{NP}$-complete

OUTLINE. This course is organized under the following units:

Core material. We study and practice the basic and advanced tools of understanding, composing, and analyzing algorithms.

- We review topics of algorithm analysis and correctness proof that students have seen in previous courses and make students' skills in these areas stronger and more rigorous.
- We learn analysis concepts and strategies such as asymptotics and complexity classes (big-theta, etc), recurrences and substitution, the master method, and amortized analysis.
- We learn algorithmic strategies such as divide and conquer, dynamic programming, and greedy algorithms.
- We study sorting algorithms at a level of rigor beyond those in earlier courses; some ideas of computational complexity are anticipated as we consider the lower bound for complexity of comparison-based sorting.

Topical material. The course will include a variety of specific algorithmic topics. Topics are drawn from

- In-depth revisiting of CSCI 345 topics
- Fibonacci heaps
- All-pairs shortest paths
- Maximum flow
- Linear programming
- Fast Fourier transform
- String matching
- Computational geometry

Computational complexity. This course plays a second duty: it provides advanced coverage of the theory of computation. Specifically

- A review of formal languages and automata
- Turing machines
- Undecidability
- Computational complexity; $\mathcal{P}$ vs $\mathcal{NP}$
- $\mathcal{NP}$-completeness, including reductions, and approximation algorithms.

For a schedule, see the course website.

Course procedures

HOW WE DO THIS COURSE. Of the courses I teach, this is the most textbook-dependent (besides a course for which I wrote the textbook) and the least lecture-oriented. Class time is used more for collaborative practice than for presentation of new material.

For most class periods, students will have a textbook reading and practice problems; in class we will clarify points in the readings, review solutions to practice problems, and work together on new problems. Students will have more serious problems sets (for turn-in and grading) about every week-and-a-half. There will be three mostly-noncumulative tests (two held during class time, the last during the exam block).

PROBLEM SETS. The bulk of the work in this course will be pencil-and-paper(-and-computer) problem sets. You are permitted to work together to a reasonable extent. If you work with another person or two, be your own judge that you are making a grade-worthy contribution and receiving the full benefit of the exercise.

Often the assignment will have special instructions for certain problems. For problems that involve writing an algorithm, the instructions will be something like this: implement your solution in a programming language of your choice; write test cases (such as JUnit or PyUnit tests) that demonstrate the solution's correctness; write a correctness proof for the algorithm; and formally analyze the algorithm's complexity, giving a big-Oh, or, preferably, a big-Theta category.

Assignments should be turned in electronically (source code for code solutions, pdf for everything else). I strongly encourage you to write up your solutions using LaTeX. I will provide some help (such as LaTeXsource for some of my notes and solutions) to get you started learning LaTeX.

Course notebook. Please obtain a notebook to use for daily work and in-class problems. Please keep it presentable so that the instructor can inspect it for evaluating daily work and so that your classmates can see your work in it for collaboration.

Grading. There will be three mostly-noncumulative tests (scheduled for classtime Sept 30, classtime Oct 28, and the final exam block, Wednesday, Dec 14, 10:30 am).

instrument	*weight*
Test 1	20
Test 2	20
Test 3	20
Problem sets	30
Participation/daily work	10

Note that test 3, held during the final exam block, is not worth more than the other two tests. The heavy weight given to problem sets reflects how they demand most of the work in this course.

Policies etc

Academic Integrity. Collaboration among students in the class is permitted on most assignments. Many problems you will be assigned are discussed or solved in the computer science literature or on the Internet. Use of these resources for graded work (that is, problem sets as opposed to daily work) is discouraged, and if they are ever used, full citation must be given, just as in a research paper. *Use of any resources that specifically serve as solutions to exercises in Cormen et al is not permitted.* (One exception: Solutions posted on the website that accompanies the textbook may be used to check answers for daily work; I will not assign any problems in the problem sets that have solutions posted there.)

Assignments. Graded homework will not be accepted late. If an assignment is not complete by the deadline, turn in what you have at that time for partial credit.

Attendance. Students are expected to attend all class periods. It is courtesy to inform the instructor when a class must be missed.

Examinations. Students are expected to take all tests, quizzes, and exams as scheduled. In the case where a test must be missed because of legitimate travel or other activities, a student should notify the instructor no later than one week ahead of time and request an alternate time to take the test. In the case of illness or other emergency preventing a student from taking a test as scheduled, the student should notify the instructor as soon as possible, and the instructor will make a reasonable acomodation for the student. The instructor is under no obligation to give any credit to students for tests to which they fail to show up without prior arrangement or notification in non-emergency situations. The final exam block is Wednesday, Dec 14, at 10:30 AM. I do not allow students to take finals early (which is also the college's policy), so make appropriate travel arrangements.

Special needs. *Institutional statement:* Wheaton College is committed to providing reasonable accommodations for students with disabilities. ? Any student with a documented disability needing academic adjustments is requested to contact the Academic and Disability Services Office as early in the semester as possible.? Please call 630.752.5941 or send an e-mail to? `jennifer.nicodem@wheaton.edu` for further information.

My own statement: Whenever possible, classroom activities and testing procedures will be adjusted to respond to requests for accommodation by students with disabilities who have documented their situation with the registrar and who have arranged to have the documentation forwarded to the course instructor. Computer Science students who need special adjustments made to computer hardware or software in order to facilitate their participation must also document their needs with the registrar in advance before any accommodation will be attempted.

GENDER-INCLUSIVE LANGUAGE. The college requires the following statement to be included on all syllabi: *For academic discourse, spoken and written, the faculty expects students to use gender inclusive language for human beings.*

OFFICE HOURS. I try to keep a balance: Stop by anytime, but prefer my scheduled office hours. Any time my door is closed, it means I'm doing something uninterruptible, such as making an important phone call. Do not bother knocking; instead, come back in a few minutes or send me an email.

DRESS AND DEPORTMENT. Please dress in a way that shows you take class seriously—more like a job than a slumber party. (If you need to wear athletic clothes because of activities before or after class, that's ok, but try to make yourself as professional-looking as possible.) If you must eat during class (for schedule or health reasons), please let the instructor know ahead of time; we will talk about how to minimize the distraction.

ELECTRONIC DEVICES. Please keep laptops and all electronic devices put away and silenced during class. (That's right, this is a computer science course, but you're not allowed to use a computer during class. Trying out programming concepts on your own during class time (for example) is not productive because it takes you away from class discussion. You cannot multitask as well as you think you can.) ***Text in class and DIE.*** I take this very seriously.