CSE 441T/541T Advanced Algorithms **August 26, 2009**

Course Overview and General Information

Your primary source for class information, homeworks, labs, and handouts is the class web site, **http://classes.cec.wustl.edu/∼cse441**. Please check this site regularly for course announcements.

- **Where and When**: Mondays and Wednesdays 1:00-2:30 PM in Cupples II 200; weekly recitation/studio sessions at a time and place to be announced.
- **Prerequisites**: CSE 241, or equivalent undergraduate training in algorithms and data structures, complexity analysis, and basic proof techniques.
- **Your Instructor**: Dr. Jeremy Buhler. I would prefer that you use the course address, *cse441@cec.wustl.edu* rather than my personal address, *jbuhler@cse.wustl.edu*, for course-related correspondence. See the section on Contact Info below.
- **Your Teaching Assistant**:
  - Alan Kwan (*alan@ural.wustl.edu*)

  In addition, David Lu!! will sometimes act as facilitator for the recitation/studio sessions.
- **Contact Info**: You can contact the instructor and/or TA by any of the following means:
  - Send mail to the course account, *cse441@cec.wustl.edu*. Your message will be seen by the instructor and TA.
  - Send mail to the instructor, *jbuhler@cse.wustl.edu*. Please use the cse441 account in preference to my address if at all possible.
  - Drop in to see us during our office hours, or by appointment. My office is Jolley 530. To call for an appointment, my phone number is 314-935-6180 (though I prefer email). The TA holds office hours as indicated on the office hours page of the website.
- **Office Hours**: See the course website.
- **Textbook**: Cormen, Leiserson, Rivest, and Stein's *Introduction to Algorithms*, 2nd Edition. The first edition is not sufficient because I may assign problems out of the book that only appear in the second edition.

1 Course Philosophy and Structure

This course is about the design and analysis of algorithms, with a strong emphasis on *optimization.* That is, given a problem with (possibly) a multitude of feasible solutions and a measure of how good each feasible solution is, we want to find the *best* solution. The problems will be *combinatorial* in nature; that is, they will have discrete (as well as possibly continuous) components.

The only way I know to achieve proficiency in solving combinatorial optimization problems is through experience, i.e. by thinking your own way to a solution. My goals for our regular class meetings are to point out some basic structures that are common to large classes of optimization problems and, with your help, to work through examples that show how these structures can be exploited to produce efficient algorithms (or to recognize that no such algorithms exist).

Seeing a solution is only half the battle in algorithm design; you must also express your solution in a form that clearly communicates it and *convinces others (and yourself!) of its correctness.* Thes standard of evidence for correctness in this course is a *formal proof*, just as you would encounter in any upper-level mathematics course. Part of my goal for this course is for everyone to become competent at explaining your algorithms in the language of formal proof.

In past years, I've tried to teach how to do formal proofs mainly by example and by one-on-one interactions with students who are not confident of their proof skills. While I will continue to pursue both of these approaches this year, we will also be trying a third, supplementary approach: a *proof studio*, held approximately weekly and run by my graduate TA, Alan Kwan. Attendance at studio meetings, which will run 1-1.5 hours, is not mandatory but *is* highly recommended, especially if you are looking to build your proof skills. In studio sessions, participants will first divide into small groups to work on combinatorial optimization problems. Following this small-group work, there will be an opportunity for a subset of groups to present their correctness proofs and for others to critique them for correctness, clarity, and structure.

In addition to all the above, I will supply you with some of worked examples for each major chunk of the course in the form of "practice problems." The problems and their solutions may be downloaded separately from the course web site. *Please try to work these problems yourself* before looking at the solutions – they are the best way to build and check your understanding. If you come seeking help with the homework, the first thing I'll probably ask is how you've fared with the practice problems.

You should expect to spend at least 10-14 hours on each homework, including time to work the practice problems. For each homework problem, you will need to understand what is being asked, see *how* to apply the basic structures (e.g. greedy choice, dynamic programming) that you have learned, devise an efficient solution, and write both a clear, concise description of your solution *and* a formal proof justifying both its correctness and its time complexity. *Please start early* on the homeworks. Be prepared to put aside some of the problems and come back to them. Steady mental effort, perhaps spaced over a period of hours or days, is usually rewarded.

2 CSE 441 vs. CSE 541

This course is actually *two courses* in one (yes, it's both a floor wax *and* a dessert topping!). Both courses share the same set of lectures and much of the homework and exam material. However, students taking the more advanced course, CSE 541, will be expected to complete additional, more challenging homework and exam problems. Students taking 441 are not required to do the extra problems but may choose to do them for extra credit.

Anyone may sign up for either CSE 441 or CSE 541. If you sign up for 441, you may switch to 541 up until the due date for the first homework. If you sign up for 541, you may switch to 441 any time up until the late drop deadline (November 13th).

3 Homeworks

There will be four to five homework assignments. Assignments will be distributed in PDF form from the course web page.

Assignments must be turned in **at the beginning of class on the due date** or placed in a turn-in box outside Jolley 530 by 12:30 PM on the due date. Assignments turned in anywhere besides the above locations (including my mailbox) will not be counted unless prior arrangements have been made with me, since I won't be able to tell whether homeworks appearing in random locations were turned in on time. Homeworks turned in at the end of class will be accepted or rejected at the instructor's whim (seriously, I will decide on a policy for this once I see whether people are skipping class to do their homeworks).

Do not assume that I will bring a stapler to class, or that your favorite printer will be working five minutes before class time. Late homeworks **will not** be accepted, as I plan to hand out solutions on the day that they are due. I expect your solutions to be clear, concise, neat, and easy to read – if the graders and I cannot understand your argument, we will mark it wrong. Time spent typesetting your homework in, e.g., LaTeX or Word is much appreciated and will also make it easier for you to make corrections as you're writing the solutions.

If you feel your assignment was graded incorrectly, please take the matter up with the TA before talking to me. My TA works hard, and I prefer to support his decisions.

4 Exams and Grading

There will be two 80-minute exams held in class during the semester, plus one cumulative final exam given during the finals period.

Your final grade in the course will be weighted roughly as follows:

1. each homework: 12.25% (assuming four)
2. each exam: 17%

Exams are always closed-book and closed-notes. However, you may use one 8.5×11 crib sheet (both sides) if desired.

I will be asking for student volunteers to help grade the homeworks. If you have solved a homework problem and want to grade it, you must write up your solution and explain it to me *before* its due date. Graders will be chosen on a first-come, first-served basis. Those chosen to grade will mark the homeworks at an evening grading meeting run by the TA. Due to the difficulty of grading 40+ assignments in a timely fashion, I usually need *two* student graders per problem.

Depending on how well I feel you did at grading a problem, you may receive extra credit up to the value of the problem you grade. Each student may volunteer to grade one problem (on one problem set) during the semester. I will try to avoid repeat grading; however, I can only spread the grading credit fairly if everyone takes the time to apply for grading and to provide me with good solutions in advance!

5 Policy on Collaborations and Academic Integrity

Please see the separate document on this subject on the course web site. You are expected to be familiar with this document and to attest (by your signature) that you have followed the course collaboration policy on each homework you turn in.