ORIE 6334 Approximation Algorithms

Course Information

September 1, 2009

1 Instructor Information

Instructor:	David Williamson
Office:	Rhodes 236
Office hours:	Mondays 1-2, Wednesdays 11-12, and by appointment.
Office phone:	255-4883
Email:	`dpw@cs.cornell.edu`

2 Lectures

Lectures will be held in Hollister 320 on Tuesdays and Thursdays from 1:10-2:25.

3 Course website

The course website will be at `www.orie.cornell.edu/~dpw/or6334/index.html`. Various materials from the course will be posted there.

4 Prerequisites

There is no formal prerequisite. In practice, I will be assuming some previous exposure either to algorithms or combinatorial optimization, and some ability to do mathematical proofs. If you've had a good undergraduate algorithms class that had proofs about the algorithms, you should be set. Please talk to me if you have questions about whether you have the necessary background.

5 Textbooks

The required text is "The Design of Approximation Algorithms", which I am in the process of writing along with David Shmoys. A draft of the book is available as a coursepack from the Cornell Store for $25. If they are out of copies, ask at the desk for them to print another copy for you. Coverage of the material in class will follow the book closely, and problem sets will (mostly) be assigned from the exercises at the end of each chapter.

Other materials that might be helpful for the course:

- Vijay V. Vazirani, *Approximation Algorithms*, Springer, 2004. A good text.

- Bernhard Korte and Jens Vygen, *Combinatorial Optimization*, 3rd edition, Springer, 2005. A reasonably good-sized section on approximation algorithms with some material not covered by Vazirani.

- Dorit S. Hochbaum, editor, *Approximation Algorithms for NP-hard Problems*, PWS Publishing Company, Boston, 1997. A collection of surveys. Now slightly dated, but some surveys are quite good.

6 Special offer

Just for you, this semester only! As mentioned above, the course textbook is in draft form, and is scheduled to be sent off to the publisher in late January 2010. This means we are eager to find any remaining typos and mistakes, and we are interested in suggestions on how to improve presentation and wording. We are further interested in suggestions for additional exercises.

If you bring to our attention any bugs or make any suggestions that we end up using, your name will be immortalized[1] in the preface. *But wait, there's more!* I will pay \$1 to the first person who brings a typo or mistake to my attention[2]. Similarly, I will pay \$1 to the first person making a particular suggestion that we end up implementing (e.g. suggested rewording, change in presentation, exercise). I reserve the right to cap the total amount I pay out to any one person or to students in the course overall. You must be signed up in the course (either for credit or as an auditor) to be eligible for the cash.

7 Requirements

There will be 5 problem sets, handed out and collected on a biweekly basis. In addition, there will be a take-home final, which will count for somewhat more than a problem set.

8 Collaboration

Cornell's Code of Academic Integrity can be found at `cuinfo.cornell.edu/Academic/AIC.html`.

Your work on problem sets and exams should be your own. You may discuss approaches to problems with other students, but as a general guideline, such discussions may not involve taking notes. You must write up solutions on your own independently, and acknowledge anyone with whom you discussed the problem by writing their names on your problem set. You may not use papers or books or other sources (e.g. material from the web) to help obtain your solution.

No collaboration will be allowed for the take-home final.

[1] As long as the book is in print, or until someone proves that $P = NP$, whichever comes first.

[2] Knuth pays \$2.56, but he's probably richer and is certainly less error-prone than I am.

9 Schedule

Here is a rough schedule for the course, which will is subject to change without notice. Depending on the background of the class, I may go either faster or slower than this schedule indicates.

Sept	1, 3	Introduction to approximation algorithms: Set cover. Problem set 1 out.
Sept	8, 10	Greedy algorithms and local search: Bank float. Minimum degree spanning trees.
Sept	15, 17	Rounding data and dynamic programming: Knapsack. Bin packing. Problem set 1 due. Problem set 2 out.
Sept	22, 24	Deterministic LP rounding: Bin packing. Facility location.
Sept Oct	29 1	Random sampling and randomized LP rounding: MAX SAT. Facility location. Prize-collecting Steiner tree. Problem set 2 due. Problem set 3 out.
Oct	6, 8	Semidefinite programming: MAX CUT. Quadratic programming. Coloring 3-colorable graphs.
Oct Oct	10 13	No class (Fall break). Cuts and metrics: Multicut.
Oct	20, 22	Cuts and metrics: Tree metrics. Problem set 3 due. Problem set 4 out.
Oct	27, 29	Primal-dual method: Feedback vertex set. Shortest path. Generalized Steiner tree.
Nov	3, 5	Primal-dual methods: Facility location. k-median via Lagrangean relaxation. Problem set 4 due. Problem set 5 out.
Nov	10, 12	Greedy algorithms and local search revisited: k-median and facility location.
Nov	17, 19	Deterministic LP rounding revisited: Min-cost bounded-degree spanning tree. Survivable network design. Problem set 5 due.
Nov Nov	24 26	Cuts and metrics revisited: Sparsest cut and low-distortion embeddings. No class (Thanksgiving).
Dec	1, 3	Cuts and metrics revisited: Oblivious routing. Minimum bisection. Take-home exam out (tentative)
Dec	8	Make-up class: TBA Take-home exam due (tentative).

10 Your information

Please fill out the information below and return it by the end of the lecture.

Name ____

Net ID ________________ Preferred email address ________________

Major ____________________ Year ____________________

Are you taking this class for credit? ____

I have studied the following subjects:

____ Linear programming
____ Algorithms
____ Combinatorial optimization
____ Complexity theory (including NP-completeness)
____ Linear algebra
____ Combinatorics and/or graph theory

I have already studied some amount of approximation algorithms (Yes/No; if Yes, how much):